

sintax

para usuarios TIMEX SINCLAIR - TK 833 y 835



Nº 3

oct-1984

Indice

Ecuaciones Diferenciales	4
Scroll	5
Galaxia	6
Vortice	7
Golf	8
Aritmetica Binaria	10
CC-Graf	12
3D-Graf	13
Bombas	14
Costos de Memoria	16
Escarabajo	23
Maquina de Frutas	24
Patrones Universales	25
Carrera	26
Cuentas Personales	28
15 Pulsos	35
Solitario	36
Tren de Multiplicacion	38
Programando con SINTAX	39
Clasificacion Alfabetica	42

ECUACIONES DIFERENCIALES

```

10 REM EC-DIFF
ECUACIONES DIFERENCIALES
MAGNASCO, M.O. 1984

20 PAPER 0: INK 7: BORDER 1: P
RINT "" Resolucion numerica de
una ecuacion diferencial de 2o
orden "" x""+a(x',t)+b(x,t)=h(t)
""
100 INPUT " Entre x"" en funcion
de x, x' y t x""(X,X1,T)="" F$
110 INPUT " Entre t en funcion
de x"" t(X2)="" T$
120 INPUT " Condiciones iniciales
x0="" X, "x'0="" X1
130 CLS : PRINT " Desea el graf
ico de x en fun- cion del tiempo
o, en el espacio de las fases,
o solo la impre- sion de los va
lores? "" Marque G para grafico
, F para fases o I para impresio
n ""
140 LET i$=INKEY$: IF i$="" THE
N GO TO 140
150 IF i$<>"F" AND i$<>"G" AND
i$<>"I" THEN PRINT AT 8,16;i$;"?
"" GO TO 140
160 LET GRA=0: LET IMP=0: LET F
ASE=0: IF i$="G" THEN PRINT AT 8
,16,"GRAFICA": LET TT=0: LET GRA
=1: INPUT "ET": ET, "EX": EX: CLS :
DRAW 0,175: PLOT 0,88: DRAW 255
170 IF i$="F" THEN PRINT AT 8,1
6;"FASES": INPUT "EX": EX, "EX'": E
X': CLS : LET FASE=1: PLOT 0,88:
DRAW 255,0: PLOT 128,0: DRAW 0,1
75
180 LET T=0: IF i$="I" THEN LET
IMP=1: PRINT AT 8,16;"IMPRIMIR"
BEEP .1,0: CLS
190 LET X2A=VAL F$: REM

200 LET DT=VAL T$: LET X2=VAL F
$: LET X=X+X1*DT+(4*X2-X2A)*DT*D
T/6: LET X1=X1+(3*X2-X2A)/2*DT:
LET T=T+DT: LET X2A=X2
210 REM

220 IF FASE THEN PLOT X*EX+128,
X1*EV+88
230 IF IMP THEN PRINT T:TAB 5:X
:TAB 20:X1
240 IF GRA THEN LET TT=TT+ET*DT
PLOT TT,X*EX+88: IF TT>=253 TH
EN CLS : DRAW 0,175: DRAW 0,-88:
DRAW 255,0: LET TT=0
250 GO TO 200

```

Es un programa para aproximar la solucion de una ecuacion diferencial de segundo orden, $F(x, x', x'', t) = 0$, donde x'' puede ponerse en funcion del resto de los parametros, $x'' = F(x, x', t)$ y doy x_0, x'_0 . El metodo es muy simple: desarrollo x en una serie de Taylor alrededor de 0:

$$x(\Delta t) = x_0 + x'_0 \Delta t + (x''_0 \Delta t^2)/2 + (x'''_0 \Delta t^3)/6 \quad x''_0 = F(x_0, x'_0, 0)$$

$$x'(\Delta t) = x'_0 + x''_0 \Delta t + x'''_0 (\Delta t^2)/2 \quad x'''_0 = -(x''_0 - x''(\Delta t)) / \Delta t$$

$$x(\Delta t) = x_0 + x'_0 \Delta t + (4x''_0 - x''') \Delta t^2/6$$

Obtenido $x(\Delta t)$, $x'(\Delta t)$, hago un desarrollo identico tomando como origen Δt , y asi sucesivamente. Esta aproximacion es correcta hasta el segundo orden en Δt .

Ejemplo:

$$x'' = -x - 0.1 \cdot x'$$

$$\Delta t = "0.02"$$

$$x_0 = 60, x'_0 = 0$$

$$\text{rango de } x = 1$$

$$\text{rango de } x' = 1$$

LISTADO 2

```

10 LET A$="50"
20 LET A$(2)=STR$ (VAL A$(2)+.
30 IF A$(2)="5" THEN GOTO 20
40 FOR I=1 TO 22
50 PRINT "===USUARIO Y PROG. C
E SINCLAIR=== "
60 NEXT I
70 FOR I=1 TO 32
80 RAND USR VAL A$
90 NEXT I
100 CLS
110 GOTO 20

```

Es un programa con código de máquina para mover el display cruzando la pantalla en cualquiera de las 8 direcciones cardinales.

Entre el listado 1, tecleando los 126 caracteres en la sentencia REM de la línea 1. Corra el programa y el código de maquina sera pokeado por la sentencia REM .

Deletée luego de la linea 10 a la 70 y reemplacelas por la rutina de demostracion del listado 2.

Si entra el comando "GO TO 10"
la versatilidad del programa
SCROLL sera demostrada.

GALAXIA

```

50 INK 7; LET N$="GALAXIA 2000
  LET A$="R": LET B$="F": LE
T C$="V": LET S=0: LET TS=0
60 BORDER 1: PAPER 0: CLS
70 PRINT INK 7; BRIGHT 1;"-----
-----GALAXIA 2000-----"
  80 PRINT AT 13,0;"EL MAXIMO PU
NTAJE LO POSEE " : PRINT : PRINT
  N$;" EN " ;TS;" PUNTOS"
  90 PRINT AT 20,0; FLASH 1;"Pul
se cualquier tecla para comenzar
"
101 PRINT AT 3,0;"Pulsando las
flechas del cursor mueve su n
ave,cuando tiene centrada en
la mira la nave invasora pu
lse 0 (CERO) para disparar el
laser"
140 PAUSE 0
150 REM INICIO
160 CLS
170 PRINT AT 8,14,B$
180 PRINT AT 13,14,C$
190 LET Q=INT (RND*16)+2: LET R
=INT (RND*26)+2
200 PLOT 0,15: DRAW 255,0: FOR
X=0 TO 40 STEP 4: PLOT 125-X,15:
DRAW -2*X,-15: PLOT 125+X,15: D
RAW 2*X,-15: NEXT X: LET Q=13: F
OR X=73 TO 125 STEP 5: PLOT X,15
: DRAW -X,-Q: LET Q=INT (Q/6)*5
NEXT X: LET Q=13: FOR X=178 TO
255 STEP 5: PLOT X,15: DRAW 255-
X,-Q: LET Q=INT (Q/6)*5: NEXT X
220 FOR I=500 TO 0 STEP -1
230 PRINT INK 7; BRIGHT 1; AT Q
R;A$
240 LET U=Q: LET V=R
250 LET Q=Q+(INT (RND*2)-1): LE
T R=R+(INT (RND*2)-1)
260 LET R=R+(2*(INKEY$="S")-(IN
KEY$="B"))
270 LET Q=Q+(2*(INKEY$="7")-(IN
KEY$="6"))
280 IF Q<=3 THEN LET Q=Q+3
290 IF R<=3 THEN LET R=R+3
300 IF Q>=18 THEN LET Q=Q-3
310 IF R>=28 THEN LET R=R-3
320 LET I=6: IF Q>9 AND Q<12 AN
D R>12 AND R<15 THEN LET I=2

```

```

330 PRINT INK 7; AT 8,14;B$; AT 1
3,14;C$
340 IF INKEY$="0" THEN GO TO 45
0
350 IF S<=0 THEN LET S=0
360 PRINT AT U,U;" " : INK 1; P
APER 6; AT 0,0;" Puntuacion=" ;S;
" : INK 0; PAPER 5;" Tiempo=" ;T
"
370 BEEP .0025,5: BEEP .0025,6
380 NEXT T
390 GO SUB 650
400 PRINT INK 6; FLASH 1; AT 10
10;"FIN": INK 2;" Puntos=" ;S
410 IF S>TS THEN GO SUB 600

```




```

420>BORDER 1: PRINT AT 20,2: IN
0:"Presione cualquier tecla pa
" jugar de nuevo."
430 LET S=0
440 FOR X=0 TO 500
450 IF INKEY$("<>") THEN GO TO 60
460 NEXT X
470 GO TO 50
480 BEEP .01,7: FOR P=2 TO 6 ST
490 INK P: PLOT 10,15: DRAW 11
500 DRAW 120,-72: PLOT 11,15
510 112,71: DRAW 120,-71: NEXT
P
520 BEEP .015,9: BEEP .015,8
530 IF Q>9 AND Q<12 AND R>12 AN
540 R<16 THEN GO TO 530
550 LET S=S-20
560 GO TO 580
570 PRINT PAPER 6: INK 2: FLAS
580 AT Q,R-1:"":AT Q+1,R:""
590 AT Q-1,R:""
600 FOR X=0 TO 10: BEEP .005,4
610 BEEP .0,9: NEXT X
620 BEEP .01,7: BEEP .015,2
630 FOR X=0 TO 50: NEXT X: PRIN
640 AT Q,R-1:"":AT Q-1,R:""
650 AT Q+1,R:"": LET Q=INT (RND*15
660 +2): LET R=INT (RND*26)+2
670 LET S=S+50
680 FOR P=19 TO 11 STEP -1: PRI
690 NT AT P,0:"": NEXT P
700 GO TO 350
710 FOR X=0 TO 150: : : : BORDE
720 R: BORDER 6: NEXT X: INPUT "F
730 LASH 1:"MAXIMA PUNTUACION "; FLA
740 SH 0:"Escribe tu nombre (max.9 l
750 etras)"; LINE n$
760 IF LEN n$>10 THEN GO TO 60

```

NOTA GRAFICA:

Para dibujar la nave: GRAPHICS "AB". Los graficos de B\$ son: GRAPHICS "D E", y los de C\$ son GRAPHICS "F G".

```

620 LET T5=5
630 FOR X=0 TO 15: BORDER RND*7
640 FOR U=0 TO 9: NEXT U: NEXT X
650 GO TO 420
660 FOR N=0 TO 2: BEEP .2,0: BE
670 .3,5: NEXT N: BEEP .3,2: BEEP
680 .2,5: BEEP .3,6: BEEP .2,7: BEE
690 .2,6: BEEP .2,5: BEEP .2,4
700 RETURN
710 RESTORE 680: FOR x=0 TO 55:
720 READ a: POKE USR "a"+x,a: NEXT
730 RUN
740 DATA 64,135,136,243,243,136
750 135,64,2,225,17,207,207,17,225
760 174,0,181,0,146,0,170,73,254,15
770 160,144,136,128,128,0,127,3,5
780 9,17,1,1,0,0,128,128,136,144,160
790 192,254,0,1,1,17,9,5,3,127

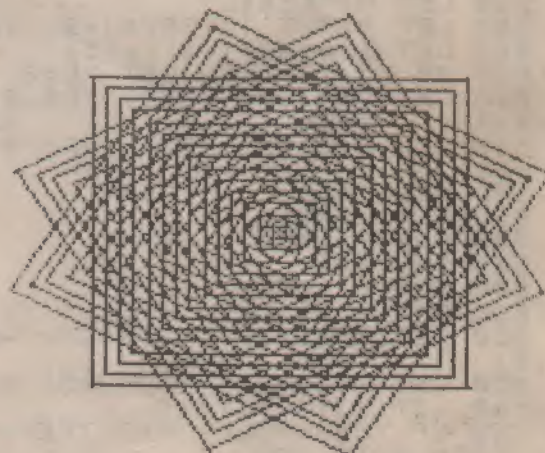
```

VORTICE

```

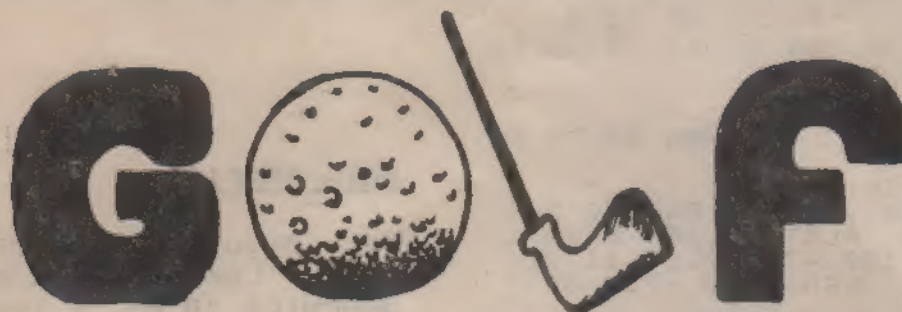
10 REM VORTICE
20 MAGNESCO M. O. 1984
30 CLS: PRINT "Este program
40 ma una figura geometrica reg
50 lar y la rota y disminuye sobr
60 la pantalla."
70 INPUT "Numero de lados de l
80 a figura: "; N: INPUT "Angulo de
90 rotacion: "; F
100 CLS: PRINT N: AT 21,0: F: LE
110 C=2*PI/N: LET F=F*PI/360
120 IF N=3 THEN GO TO 200
130 FOR R=85 TO 0 STEP -2: LET
140 A=R*F
150 FOR I=0 TO N-1: LET xa=r*cos
160 (a+i*0): LET ya=r*sin (a+i*0)
170 PLOT xa+128,ya+88: DRAW r*cos
180 (a+(i+1)*0)-xa,r*sin (a+(i+1)*0)-
190 ya: NEXT I
200 NEXT R
210 STOP
220 FOR R=1 TO 65 STEP 2: LET A
230 =R*F: LET X1=R*cos (A+0): LET Y1
240 =R*sin (A+0): LET X2=R*cos (A+0+
250 C): LET Y2=R*sin (A+0+C): LET X3
260 =R*cos (A+0+C+C): LET Y3=R*sin (
270 A+0+C+C): PLOT X1+128,Y1+88: DRA
280 W X2-X1,Y2-Y1: DRAW X3-X2,Y3-Y2:
290 DRAW X1-X3,Y1-Y3: NEXT R

```



Es un programa que dibuja una figura geometrica regular, la rota y la disminuye de tamaño. Tiene dos partes: una parte para dibujar cualquier poligono, y otra especificamente para triangulos.

Ejemplo: 3, 8°



```

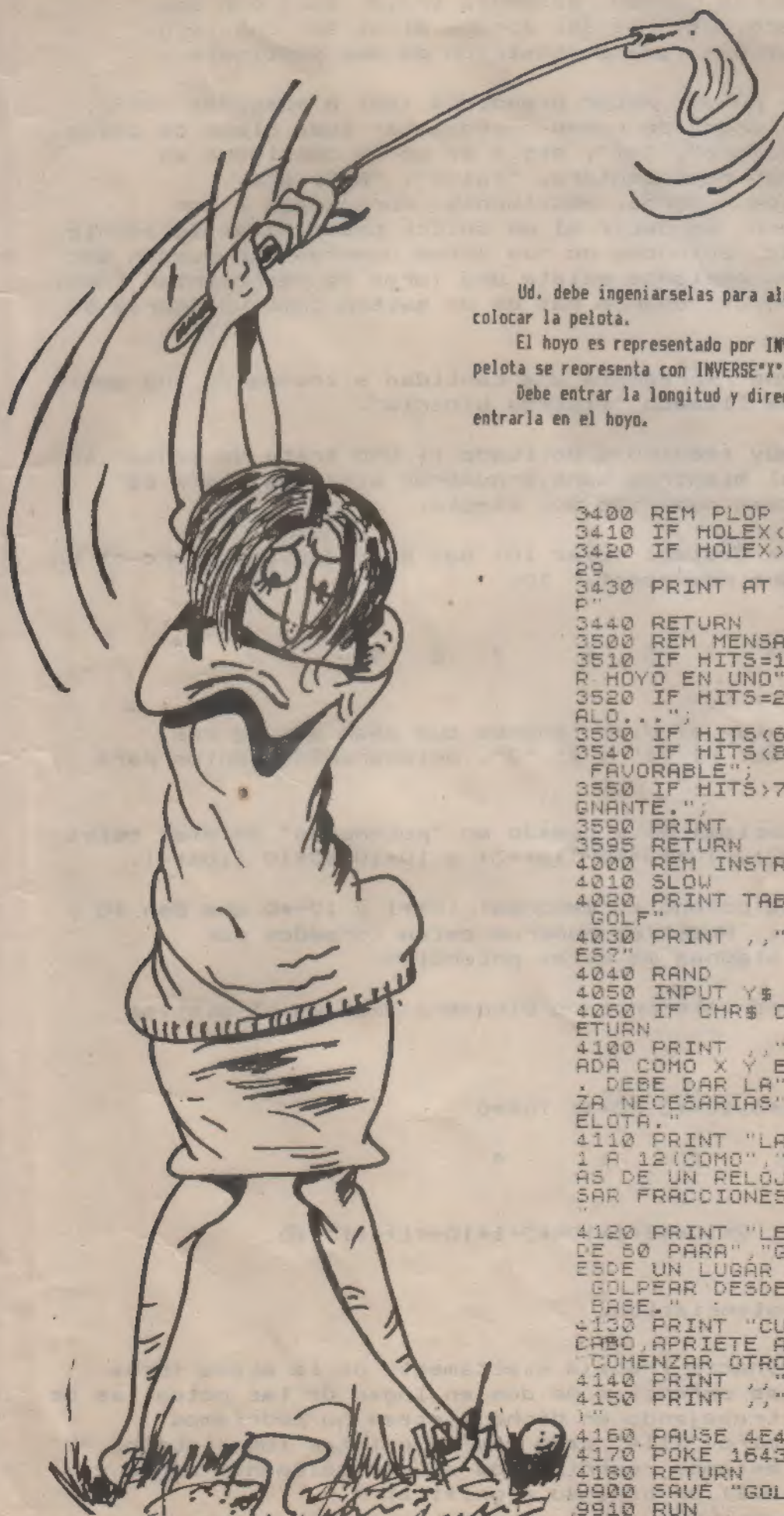
1 REM *** GOLF***
10 GOSUB 4000
20 CLS
30 REM DIBUJE BORDES
40 FAST
50 GOSUB 2000
100 LET HOLEX=INT (RND*30)+1
110 LET HOLEY=INT (RND*18)+1
120 LET TEEEX=INT (RND*30)+1
130 LET TEEY=INT (RND*18)+1
140 IF TEEEX=HOLEX AND TEEY=HOLEY THEN GOTO 120
150 LET HITS=0
190 SLOW
200 PRINT AT HOLEY,HOLEX;"O"
210 PRINT AT TEEY,TEEEX;"X"
300 REM CURVA PPAL.
310 GOSUB 3100
320 PRINT "DIRECCION?"
330 INPUT XD
340 GOSUB 3000
350 IF XD<0 OR XD>12 THEN GOTO 300
360 PRINT TAB 22;XD
370 GOSUB 3000
380 PRINT "INTENSIDAD?"
390 INPUT XS
400 LET XS=INT XS
410 GOSUB 3000
420 PRINT TAB 28;XS
500 LET A=TEEEX*2
510 LET B=INT ((TEEY*(-2))+42)
520 LET XD=(XD*(-1))+15
530 IF XD=12 THEN LET XD=0
540 LET C=A+INT (COS ((PI/6)*XD)*XS)
550 LET D=B+INT (SIN ((PI/6)*XD)*XS)
560 LET XA=A
570 LET XB=B
580 GOSUB 1500
590 LET P=1
600 GOSUB 1000
610 PRINT AT TEEY,TEEEX;" "
620 LET TEEY=INT ((D-42)/(-2)+0.5)
630 LET TEEEX=INT ((C+0.5)/2)
640 LET HITS=HITS+1
700 IF TEEEX=HOLEX AND TEEY=HOLEY THEN GOTO 800
710 LET A=XA
720 LET B=XB
730 LET P=0
740 GOSUB 1000
750 GOTO 200
800 GOSUB 3400
810 PRINT AT 20,0;"UD. LO HIZO"
820 PRINT "LE LLEVO";HITS;"LANZAMIENTOS"
830 GOSUB 3500
840 IF INKEY$="" THEN GOTO 840
850 CLEAR
860 GOTO 20
1000 LET U=C-A
1010 LET V=D-B
1020 LET D1X=SGN U
1030 LET D1Y=SGN V
1040 LET D2X=SGN U
1050 LET D2Y=0

```

```

1060 LET M=ABS U
1070 LET N=ABS V
1080 IF M>N THEN GOTO 1130
1090 LET D2X=0
1100 LET D2Y=SGN V
1110 LET M=ABS U
1120 LET N=ABS V
1140 LET S=INT (M/2)
1150 FOR I=0 TO M
1160 IF P THEN PLOT A,B
1165 IF NOT P THEN UNPLOT A,B
1170 LET S=S+N
1180 IF S<M THEN GOTO 1230
1190 LET S=S-M
1200 LET A=A+D1X
1210 LET B=B+D1Y
1220 NEXT I
1230 LET A=A+D2X
1240 LET B=B+D2Y
1250 NEXT I
1260 RETURN
1500 REM PRUEBE NUEVA UBICACION PARA TEE
1510 LET ERROR=(C<2 OR C>62 OR D<6 OR D>42)
1520 IF NOT ERROR THEN RETURN
1530 PRINT AT 21,0;"DENTRO DE TERRENO ESCABROSO-GOLPE PENAL"
1540 LET HITS=HITS+1
1550 IF C<2 THEN LET C=2
1560 IF C>62 THEN LET C=62
1570 IF D<6 THEN LET D=6
1580 IF D>42 THEN LET D=42
1590 RETURN
2000 REM DIBUJE LIMITES VERDES
2010 LET Y=43
2020 GOSUB 2100
2030 LET Y=4
2040 GOSUB 2100
2050 LET X=0
2060 GOSUB 2200
2070 LET X=63
2080 GOSUB 2200
2090 RETURN
2100 REM DIBUJE LA LINEA
2110 FOR X=0 TO 63
2120 PLOT X,Y
2130 NEXT X
2140 RETURN
2200 REM DIBUJE LA LINEA"Y"
2210 FOR Y=4 TO 43
2220 PLOT X,Y
2230 NEXT Y
2240 RETURN
3000 REM LIMPIE APUNTES
3010 PRINT AT 20,0;" "
3020 PRINT AT 20,0;" "
3030 RETURN
3100 REM LIMPIE LINEA 20
3110 PRINT AT 20,0;" "
3120 PRINT AT 20,0;" "
3130 RETURN

```

Ud. debe ingeniárselas para alcanzar la verde, y luego colocar la pelota.

El hoyo es representado por INVERSE*0*, mientras que la pelota se reopresenta con INVERSE*X*.

Debe entrar la longitud y dirección requerida para entrarla en el hoyo.

```

3400 REM PLOP
3410 IF HOLEX<2 THEN LET HOLEX=2
3420 IF HOLEX>29 THEN LET HOLEX=
29
3430 PRINT AT HOLEY,HOLEX-2,"PLO
P"
3440 RETURN
3500 REM MENSAJES ABSURDOS
3510 IF HITS=1 THEN PRINT "-HACE
R HOYO EN UNO";
3520 IF HITS=2 THEN PRINT "-NO M
ALO...";
3530 IF HITS<6 THEN GOTO 3590
3540 IF HITS<8 THEN PRINT "-SOLO
FAVORABLE";
3550 IF HITS>7 THEN PRINT "-REPU
GNANTE.";
3590 PRINT
3595 RETURN
4000 REM INSTRUCCIONES
4010 SLOW
4020 PRINT TAB 7;"LANZAMIENTO DE
GOLF"
4030 PRINT ",";"QUIERE INSTRUCCION
ES?"
4040 RAND
4050 INPUT Y$
4060 IF CHR$(CODE+Y$)<>"S" THEN R
ETURN
4100 PRINT ",";"LA PELOTA ES MOSTE
ADA COMO X Y EL","HOYO COMO 0.00
. DEBE DAR LA","DIRECCION Y FUER
ZA NECESARIAS","PARA HUNDIR LA P
ELOTA."
4110 PRINT "LAS DIRECCIONES SON
1 A 12(COMO","SOBRE LAS MANECILL
AS DE UN RELOJ)PERO","UD. DEBE
SAR FRACCIONES,COMO 2.6","O 10.
4120 PRINT "LE LLEVA UNA FUERZA
DE 50 PARA","GOLPEAR LA PELOTA
DESDE UN LUGAR A","OTRO Y 36 PARA
GOLPEAR DESDE","LA CUSPIDE A LA
BASE."
4130 PRINT "CUANDO UN JUEGO SE A
CABO,APRIETE ALGUNA","TECLA PARA
COMENZAR OTRO."
4140 PRINT ",";"LO HA BURLADO."
4150 PRINT ",";"(APRIETE UNA TECLA
)"
4160 PAUSE 4E4
4170 POKE 16437,255
4180 RETURN
9900 SAVE "GOLF"
9910 RUN

```


Las computadoras pueden realmente tratar solo con dos numeros, uno y cero. Esto es asi porque ellas son una larga coleccion de switches cuya disposicion es muy particular.

Los switches pueden estar prendidos (on) o apagados (off). Si un switch esta prendido puede representar toda clase de cosas tales como, "verdadero", "si", etc.; si se lo considera en posicion de apagado representara, "falso", "no", etc.. Remitiendonos a los numeros, usualmente representan a dos numeros: uno y cero. Es decir si un switch puede estar solamente prendido o apagado, entonces no hay otros numeros que puedan ser representados. No obstante existe una forma de representar todos los restantes numeros, usando mas de un switch como una serie de unos y ceros.

El sistema que representa una cantidad a traves de una serie de unos y ceros es llamado "sistema binario".

Esto suena muy tecnico y delicado si uno trata de pensar en el sistema decimal mientras maneja numeros binarios, pero es realmente un sistema numerico muy simple.

En el sistema decimal (base 10) hay diez digitos, pero no un simbolo simple para representar 10.

0 1 2 3 4 5 6 7 8 9

Para representar el diez tenemos que usar dos de los simbolos anteriores: el "1" y el "0", colocandolos juntos para formar "10".

El sistema decimal esta basado en "potencias" de diez tales como: 10×10 (10^{**2}), $10 \times 10 \times 10$ (10^{**3}) y $10 \times 10 \times 10 \times 10$ (10^{**4}).

Otras son las potencias menores: 10^{**1} y 10^{**0} que dan 10 y 1 respectivamente. Nuestros numeros estan formados por combinaciones de algunas de estas potencias.

Veamos como es interpretado el numero 2114 en el sistema decimal:

Pot. de 10: 10^{**3} 10^{**2} 10^{**1} 10^{**0}

Simb. usado: 2 1 1 4

Lo cual significa: $2 \times 10^{**3} + 1 \times 10^{**2} + 1 \times 10^{**1} + 4 \times 10^{**0}$.

NOTA: ** indica potenciacion.

El sistema binario trabaja exactamente de la misma forma haciendo uso de las potencias de dos en lugar de las potencias de diez. Si estamos trabajando en dicho sistema no podriamos representar el numero "2114" dado que no existen los simbolos "2" y "4", es decir, estamos restringidos a usar solamente ceros y unos. El numero 10011 en binario significa:

pot. de dos:	2**4	2**3	2**2	2**1	2**0
equiv. dec.:	16	8	4	2	1
simb. a usar:	1	0	0	1	1

cuyo significado es: $1 \cdot 2^{**4} + 0 \cdot 2^{**3} + 0 \cdot 2^{**2} + 1 \cdot 2^{**1} + 1 \cdot 2^{**0}$.

Si realizamos esta suma obtenemos el valor 19 en el sistema decimal.

El sistema binario presenta dificultades para trabajarlo, pues los numeros son representados con muchos digitos es muy facil equivocarse al escribirlos.

Cada uno de esos digitos binarios se denomina BIT (Binary digIT). Cada switch puede tener un bit, agrupandolos generalmente de a ocho conformando un BYTE. Los 64 lots de 1024 BYTES de memoria en el enteramente expandido TS1000 y ZX81 contiene

$64 \cdot (1.024 \cdot 8) = 5.240.288$ de estos switches.

Estos solo forman la "random acces memory", pero hay muchos mas de estos switches en otras partes de la computadora. Por que pensar en 1024?. Por que no 100, un numero mas simple?. La respuesta esta pregunta esta relacionada con la aritmetica binaria. Fundamentalmente, muchos los numeros que tienen sentido en computacion se relacionan con este sistema de numeracion.

Normalmente pensamos en nuestro familiar sistema decimal y en numeros tales como 100 o 10.000, como numeros puros y pues son potencias exactas de 10. En el sistema binario hay tambien numeros puros, pero expresados en potencias exactas de 2 y no de 10.

Entonces, para la computadora, un numero exacto es 1024 y no 1000, ya que el primero es potencia de 2 y el segundo de 10.

La razon por la cual un byte (8 bits) no puede almacenar mas que 255, es que si los ocho switches estan prendidos, el valor representado es:

Nro. bit:	8	7	6	5	4	3	2	1
Pot. dos:	2**7	2**6	2**5	2**4	2**3	2**2	2**1	2**0
valor :	128	64	32	16	8	4	2	1

cuya suma es el valor 255 en decimal. Sumando un uno a este valor resultara un "acarreo" tal como cuando al sumar 1 a 9.999 genera un acarreo.

11111111
+ 1

100000000

9.999
+ 1

10.000

En ambos casos el acarreo significa que hay un dígito extra en el resultado. Un byte de 8 bits no puede contener un noveno bit, este bit extra provoca lo que se denomina "overflow".

Usando dos bytes por número, podemos tener 65.336 si todos los switches están en on.

1er. byte: 11111111
2do. byte: 11111111

Pero el decimosexto bit tiene un especial propósito: representar el signo más o el menos. Solamente siete de los dígitos pueden ser usados para representar el número y entonces, solamente 32.767 puede ser almacenado. Esto muestra una seria reducción pero no implica un desperdicio del sistema. Antes, el rango de números almacenables iba de 0 a 65.536 y ahora, usando el octavo bit del segundo byte como bit de signo, el rango va desde -32.767, a través de 0, hasta 32.766.

64K es el número máximo de direcciones de memoria que el procesador Z80 puede tratar. Nuestras computadoras tienen un set de caracteres con 255 códigos en él. Ese rango de 0 a 255 da un total de 256 números, número que es potencia de dos. Guardar uno cualquiera de tales números ocupa tan solo un byte.

Las cadenas de caracteres (strings) pueden tener, como máximo, 256 caracteres, por las mismas razones.

TS 2055/SPECTRUM

3D.
g
r
a
f
C.C
g
r
a
f

```

1 REM      CC-GRAF
      MAGNASC0, M 0 1984
5 IF PEEK 20746<>0 THEN GO TO
100
10 CLEAR 63205: LET U=63206: R
ESTORE READ a,b,c,d,e,f: DATA
10,11,12,13,14,15
20 LET U$="*2100603600237cfe78
20f8c9f33201d3f48bffc6fffd3ff3e90
f5f6cd820e73dbfffcbbfd3ffa1d3f4f1
fe00200332c25cfbc9"
30 FOR n=2 TO LEN U$-1 STEP 2:
POKE U,16*VAL U$(n)+VAL U$(n+1)
LET U=U+1: NEXT n
40 RANDOMIZE USR 63216
50 FOR a=63255 TO 63263: POKE
a,248: POKE a+8,255-PEEK (a+16):
POKE a+16,255: NEXT a
100 CLS: INPUT "ENTRE F(x) 10<
f:=7.41 F(x,y)=": IF# I
NPUT "Escala en x": EX "Escala en
y": EY LET ex=EX*16 LET ey=EY*
128
105 OUT 255,2: LET U=24576: GO
SUB 300
110 FOR Z=0 TO 2: FOR J=0 TO 7:
FOR I=J+64*Z TO J+64*Z+58 STEP
6: FOR K=0 TO 31: LET X=EX*(K-15)
120 LET Y=(I-96)*EY: LET in=INT
VAL f$: LET X=X+EX/2: LET pa=6*
INT VAL f$: POKE U,pa+in: LET U=
U+1: NEXT K: NEXT I: NEXT J: NEX
T Z
200 BEEP .1,50: PAUSE 0: OUT 25
5,0
250 STOP
300 FOR i=0 TO 21: PRINT AT i,0
: NEXT i: RETURN
600 FOR i=24576 TO 30719: POKE
i,0: NEXT i: RETURN

```




Son dos programas hermanos destinados a un mismo fin: graficar funciones de dos variables.

3D da una representacion tridimensional de la funcion (en perspectiva caballera). Sirve primordialmente para graficar superficies de variacion suave, como polinomios, gaussianas, etc..

Ejemplo:

$F(x,y) = \exp(-x*x-y*y)$
 . rango de $x=2$
 . rango de $y=2$

NOTA: en todo lugar de un programa donde haya que evaluar x^2 , A^3 , conviene escribir $x*x$, $A*A*A$. El motivo, es que para evaluar a una potencia, la maquina ejecuta:

$A^3 = \exp(B \cdot \ln(A))$,
 cuya evaluacion es muchamas lenta.

CC da una representacion codificada en color (estilo fotografia sintetica o centellograma) del valor de $F(x,y)$. Debido a que solo hay disponibles 8 colores, este programa no da demasiada informacion sobre gaussianas o polinomios. Sirve mas para curvas de variacion rapida cuya representacion en 3D seria confusa.

Ejemplo:

$F(x,y) = 3.6 * (1 + (\cos(x*x-y*y)))$
 variacion de $x=5$
 variacion de $y=5$

Para graficar una familia de curvas $U(x,y) = \text{cte.}$,

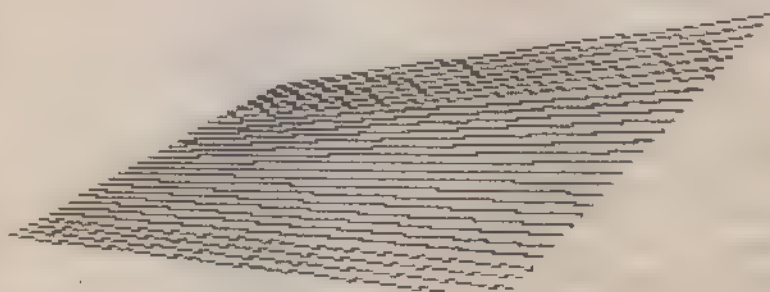
$F(x,y) = 3.6 * (1 + \cos(U(x,y)))$

donde $U(x,y) = x*x+y*y$
 $= x*x-y*y$
 $= x*x$
 $= x*x-y*y+x*y$

```

10 REM 3D GRAP MAGNASSCO, M.O. 1984
20 PAPER 0: INK 7: CLS: PRINT
  " Este es un programa destina
  do a graficar funciones de dos va
  ria-bles." " Ud debera ingresar
  la funcion, (que variara solo en
  tre 0 y 1) y dos parametros de
  escala, que definen cual es el r
  angulo que tomaran x e y.
30 OVER 1: INK 4: PLOT 96,16:
DRAW 40,0: DRAW 0,40: DRAW -40,0
DRAW 0,-40: PLOT 90,36: DRAW 8
5,0: DRAW -4,2: PLOT 116,8: DRAW
0,80: DRAW -2,-4
35 PRINT AT 15,12: PAPER 1: "
  " AT 16,12: " AT 17,12: "
  " AT 18,12: "
40 INK 7: PRINT AT 16,9: "-ex"
AT 16,16: "ex" AT 18,15: "ey" AT 2
0,15: "ey"
50 INPUT " Entre f(x,y) [ 0 <
= f <= 1 ] f(x,y) = "; f$
60 INPUT "Escala en x "; ex "Es
cala en y "; ey: LET ez=85: LET e
x=ex/75: LET ey=ey/75
100 OVER 0: CLS
110 FOR J=150 TO 0 STEP -5: LET
xa=J*.6: LET ya=J*.6: INVERSE 1
FOR i=0 TO 150 STEP 5
120 LET x=(i-75)*ex: LET y=(J-7
5)*ey: LET z=VAL f$: LET xp=i+.6
*i: LET yp=z*ez+.5*J: PLOT xa,ya
: DRAW xp-xa,yp-ya: INVERSE 0: L
ET xa=xp: LET ya=yp: NEXT i: PLO
T xp,yp: NEXT J
130 PAUSE 0: GO TO 60

```



En el REM de este programa deberá insertar 307 veces un caracter cualquiera, por ejemplo 0 (cero).

Figure 10 shows the results of the regression analysis. The dependent variable is the log of the number of patents per firm. The independent variables are the log of the number of employees, the log of the number of R&D expenditures, the log of the number of scientists, the log of the number of engineers, the log of the number of technicians, the log of the number of managers, the log of the number of salespeople, the log of the number of accountants, the log of the number of lawyers, the log of the number of other personnel, the log of the number of patents granted in the previous year, the log of the number of patents granted in the previous two years, the log of the number of patents granted in the previous three years, the log of the number of patents granted in the previous four years, the log of the number of patents granted in the previous five years, the log of the number of patents granted in the previous six years, the log of the number of patents granted in the previous seven years, the log of the number of patents granted in the previous eight years, the log of the number of patents granted in the previous nine years, the log of the number of patents granted in the previous ten years, the log of the number of patents granted in the previous eleven years, the log of the number of patents granted in the previous twelve years, the log of the number of patents granted in the previous thirteen years, the log of the number of patents granted in the previous fourteen years, the log of the number of patents granted in the previous fifteen years, the log of the number of patents granted in the previous sixteen years, the log of the number of patents granted in the previous seventeen years, the log of the number of patents granted in the previous eighteen years, the log of the number of patents granted in the previous nineteen years, the log of the number of patents granted in the previous twenty years, the log of the number of patents granted in the previous twenty-one years, the log of the number of patents granted in the previous twenty-two years, the log of the number of patents granted in the previous twenty-three years, the log of the number of patents granted in the previous twenty-four years, the log of the number of patents granted in the previous twenty-five years, the log of the number of patents granted in the previous twenty-six years, the log of the number of patents granted in the previous twenty-seven years, the log of the number of patents granted in the previous twenty-eight years, the log of the number of patents granted in the previous twenty-nine years, the log of the number of patents granted in the previous thirty years.

Case	Age	Sex	Occupation	Duration of illness	Onset	Course	Outcome
1	45	M	Teacher	10 years	1980	Progressive	Death
2	52	F	Homemaker	8 years	1982	Stable	Alive
3	60	M	Engineer	12 years	1978	Progressive	Death
4	48	F	Nurse	6 years	1984	Stable	Alive
5	55	M	Doctor	9 years	1981	Progressive	Death
6	62	F	Retired	11 years	1979	Stable	Alive
7	40	M	Student	5 years	1985	Progressive	Death
8	58	F	Manager	7 years	1983	Stable	Alive
9	65	M	Farmer	13 years	1977	Progressive	Death
10	42	F	Teacher	4 years	1986	Stable	Alive
11	50	M	Engineer	10 years	1980	Progressive	Death
12	57	F	Homemaker	9 years	1981	Stable	Alive
13	63	M	Doctor	11 years	1979	Progressive	Death
14	47	F	Nurse	6 years	1984	Stable	Alive
15	54	M	Manager	8 years	1982	Progressive	Death
16	61	F	Retired	10 years	1980	Stable	Alive
17	39	M	Student	5 years	1985	Progressive	Death
18	56	F	Manager	7 years	1983	Stable	Alive
19	64	M	Farmer	12 years	1978	Progressive	Death
20	41	F	Teacher	4 years	1986	Stable	Alive

Year	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100
1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	

1 BYTE

Todos los caracteres graficos desde el teclado se usan dentro de strings. Todas las palabras que se encuentran en el teclado y los simbolos, sin importar cuantos caracteres contengan, son entradas entre comillas.

Los corchetes y los simbolos matematicos tales como "*", "+", "/", "-" y "**" ocupan un byte cada uno. Todos los signos de puntuacion cuestan un byte pero la coma cuando es usada como un separador de impresion, cuesta 15 bytes de la memoria dedicada a la pantalla en el archivo de display.

NOT, usado en una instruccion condicional ocupa un byte.

6 BYTES

Todas las siguientes lineas de programa usan 6 bytes cada una:

10 CLS	10 CLEAR
10 CONT	10 COPY
10 FAST	10 SLOW
10 LLIST	10 LIST
10 LPRINT	10 PRINT
10 REM	10 RAND
10 RETURN	10 SCROLL
10 STOP	

7 BYTES

Estas son versiones principalmente de lineas de 6 bytes las cuales pueden aparacer con caracteres extras. La linea que imprime PI a pesar que es una funcion que se representa en dos caracteres ocupa solamente un byte de programa.

10 INPUT A	10 REM A
10 PRINT A	10 LPRINT A
10 PRINT PI	10 PRINT RDN

La ultima linea es usada frecuentemente con otras funciones y sera tratada con mas cuidado en la seccion que se refiere a 18 BYTES.

8 BYTES

10 PRINT A\$	10 INPUT A\$
10 PRINT ""	10 LET A=B
10 PRINT CHR\$ A	
10 PRINT STR\$ A	

Esta ultima linea muestra que tenemos un costo adicional de un byte para la funcion CHR\$.

9 BYTES

```

10 PRINT LEN A$    10 PRINT VAL A$
10 LPRINT "A"      10 PRINT CODE A$
10 LET A=B          10 PRINT "*"
10 PRINT A

```

Por esto podemos ver que LEN, VAL, CODE ocupan un byte cada una.

10 BYTES

```

10 LET A$=INKEY$
AND B=1 } como parte de IF/THEN
OR B>=1 }

```

11 BYTES

```

10 LET A$="" (string vacio)
10 PRINT "A";

```

Se puede ver que ";" cuesta un byte

12 BYTES

```

10 LET A$="A"
TAB (ver 20 BYTES)

```

13 BYTES

```

10 PAUSE

```

pero,

```

10 PAUSE 10   cuesta 14 bytes
10 PAUSE 100  cuesta 15 bytes
10 PAUSE 1000 cuesta 16 bytes

```

```

10 GO TO 1

```

pero,

```

10 GO TO 10  cuesta 14 bytes
10 GO TO 100 cuesta 15 bytes

```


14 BYTES

```
10 PRINT STR$ 1
```

Este punto es importante para señalar un interesante factor. Nuestras computadoras permitirán uso de nombres de variables en las líneas programas, siendo que otras computadoras insisten en conocer primero el valor. Los usuarios de TS 1000 pueden indicar GO TO A o GO SUB X y la computadora obedecerá el comando sin importarle cuando conocerá el valor de esas variables. Bajo el encabezamiento "8 BYTES" hay un ejemplo similar. La diferencia es que esa línea tiene un nombre de variable y no un número. El ahorro de memoria es de 6 bytes en 14 bytes, 43%. Esto es de veras valioso y su uso se verá cuando por diferentes circunstancias los costos de memoria sean investigados.

15 BYTES

```
10 LET A=1
```

pero,

```
10 LET A=10      cuesta 16 bytes
10 LET A=100     cuesta 17 bytes
10 LET A=1.1     cuesta 17 bytes
```

```
10 LET A=B      cuesta 9 bytes.
```

Entonces el beneficio de usar nombres de variables es mejor que usar números en un 40%.

16 BYTES

```
10 LET A=SIN 1
```

El costo es el mismo para las funciones: COS, TAN, ASN, ACS, ATN, INT, SGN, ABS, SQR, LN y EXP. Usando nombres de variables también se reducen los costos.

```
10 LET A= SIN A   usa 10 bytes
10 DIM A(1)       usa 16 bytes pero,
10 DIM A(10)      usa 17 bytes
10 DIM A(100)     usa 18 bytes
10 DIM A(1,1)     usa 24 bytes de los cuales 8 bytes son para
" ,1".
```

Compare esto con lo siguiente:

```
10 DIM (A)        usa 10 bytes
10 DIM (A,B)      usa 12 bytes
```


Cuando se trabaja en un programa con arreglos permite ocupar cinco bytes por numero si se trata de dimensiones simples. En arreglos multidimensionales se realiza el producto de los numeros entre parentesis y al resultado se lo multiplica por cinco. Esta cantidad es el numero de bytes que ocupara el arreglo en la zona de almacenamiento de variables.

10 DIM A(5,10) ocupa 256 bytes para el arreglo A

10 PEEK (1)

pero,

10 PRINT PEEK (10) usa 17 bytes

10 PRINT PEEK (100) usa 18 bytes

El tamaño de la direccion mas usual que aparece en un PEEK es de la forma:

10 PRINT PEEK (10000) y usa 20 bytes.

17 BYTES

10 DIM A\$(1) usa un byte mas que

10 DIM A(1) entonces todos los arreglos de tipo caracter usan byte mas que los arreglos numericos. Hay una diferencia importante entre los dos tipos de arreglos, los arreglos string solo ocupan un byte por caracter y no cinco.

10 DIM A\$(5,10) ocupa 50 bytes

18 BYTES

10 PRINT INT (RND*9)

Esta es la forma mas usual de usar la funcion RND. Aca tambien si se usan nombres de variables se reduce el costo.

10 PRINT INT (RND X) usa solamente 12 bytes.

10 PRINT TAB 1;"A"

pero la cantidad de bytes que usa TAB depende de la posicion donde se quiera imprimir. El costo afecta al archivo del display y no al area de memoria donde se almacenan las lineas de programa.

10 PRINT TAB X;"A" usa solo 12 bytes.

Como el costo de:

10 PRINT "A" es 9 bytes, el de "TAB X;" sera solamente de 3 bytes.

19 BYTES

El costo de subrutinas es de 19 bytes como minimo.

```
1 GO TO 9   cuesta 13 bytes
9 GO SUB 5
5 RETURN
```

todo esto cuesta 32 bytes por lo tanto la subrutina ocupa 19 bytes.

21 BYTES

```
10 POKE 1,1
```

pero,

```
10 POKE 10,1   cuesta 22 bytes
10 POKE 10000,1  cuesta 25 bytes
10 POKE 10000,10  cuesta 26 bytes
10 POKE A,A  cuesta 9 bytes
```

```
10 PLOT 1,1
```

pero,

```
10 PLOT 1,10   cuesta 22 bytes
10 PLOT 1,40   cuesta 23 bytes
10 PLOT A,A    cuesta 9 bytes
```

Todos los comandos UNPLOT tienen el mismo costo. Una o dos posiciones de ploteo le ocupan 1 byte al archivo de display por cada posicion impresa.

23 BYTES

```
10 LET A=2*2  cuestan 23 bytes pero como
10 LET A=2    cuesta 15
```

el costo del numero y el signo de multiplicacion es de 8 bytes. Un metodo alternativo para calcular raices cuadradas cuesta la misma cantidad de memoria:

```
10 LET A=2**2
10 LET A=-2*-2  cuesta 25 bytes y
10 LET A=-2**2 se reduce a 24 bytes pero da respuesta
incorrecta!!
```


Todos los numeros al cuadrado son positivos y la computadora da resultado negativo. Esto significa una falla en el lenguaje.

Si usted esta escribiendo un programa donde calcula potencias cuadradas le sera necesario incluir la funcion ABS en todas las lineas que incluyen a funcion "**".

10 FOR J=1 TO 9 cuesta 23 bytes pero,
10 FOR J=1 TO 10 cuesta 24 bytes
10 FOR J=1 TO 9 STEP 2 cuesta 31 bytes pues "STEP 2" cuesta 1 bytes.

El costo basico del loop FOR/NEXT es:

```
10 FOR J=1 TO 9
20 NEXT J
```

} 30 bytes

24 BYTES

10 IF A=1 THEN GO TO 9 pero
10 IF A=1 THEN GO TO 10 cuesta 25 bytes.

El uso de "<=", "<", ">" y "<>" en cualquier linea cuestan siempre los mismo.

10 IF A=1 OR B<2 THEN GO TO 9 cuesta 34 bytes pues "OR B<2" cuesta 10.

10 IF NOT A=1 THEN GO TO 9 cuesta 25 bytes pues "NOT" ocupa tan solo 1 byte.

10 IF A=1 THEN GO SUB 9 tambien cuesta 24 bytes

pero ella incluire al menos una linea extra GO TO para saltar esta subrutina. Si todas las subrutinas estan al final del programa la sentencia STOP hara que se reduzca su costo en unos 6 bytes.

Nuestras computadoras aceptaran nombres de variables tanto en el enunciado GO TO como en el GO SUB y seran guardadas como las tiene.

IMPRIMIENDO CON O SIN FORMATOS

PRINT AT

10 PRINT "A" cuesta basicamente 9 bytes

10 PRINT AT 1,1;"A" cuesta 26 bytes pues "AT 1,1;" ocupa 17, la mayoria de estos para los dos numeros.

10 PRINT AT L,L;"A" cuesta solamente 14 bytes

Cada posicion de impresion a lo largo de una linea cuesta un byte extra al archivo del display.

TAB

10 PRINT TAB 1;"A" cuesta 18 bytes pues "TAB 1;" ocupa 9 bytes

10 PRINT TAB X;"A" cuesta solamente 12 bytes del espacio para programa.

COMPARANDO EL COSTO DEL PRINT AT CON EL PRINT VACIO

10 PRINT AT 4,1;"A" cuesta 26 bytes pero,

10 PRINT	}	cuesta 27 bytes
11 PRINT		
12 PRINT		
13 PRINT "A"		

El precio de usar enunciados PRINT vacios para espaciar un texto es barato cuando se quiere separar una o dos lineas. Para dejar tres o mas lineas blanco es mas conveniente usar PRINT AT pues es de menor costo.

10 PRINT	}	cuesta 24 bytes
11 PRINT TAB 5;"A"		

mientras que:

10 PRINT AT 2,5;"A" cuesta 26 bytes.

10 PRINT AT X,Y;"A" cuesta solamente 14 bytes para almacenarlo en el area de programa.

STRING

10 PRINT A\$ cuesta basicamente 8 bytes
 10 PRINT A\$(TO 9) cuesta 18 bytes pues "(TO 9)" ocupa 10 bytes
 10 PRINT A\$(TO 10) cuesta 9 bytes pues hay un caracter mas
 10 PRINT A\$(1 TO 9) ocupan 25 bytes de los cuales 17 son el costo de "(1 TO 9)"

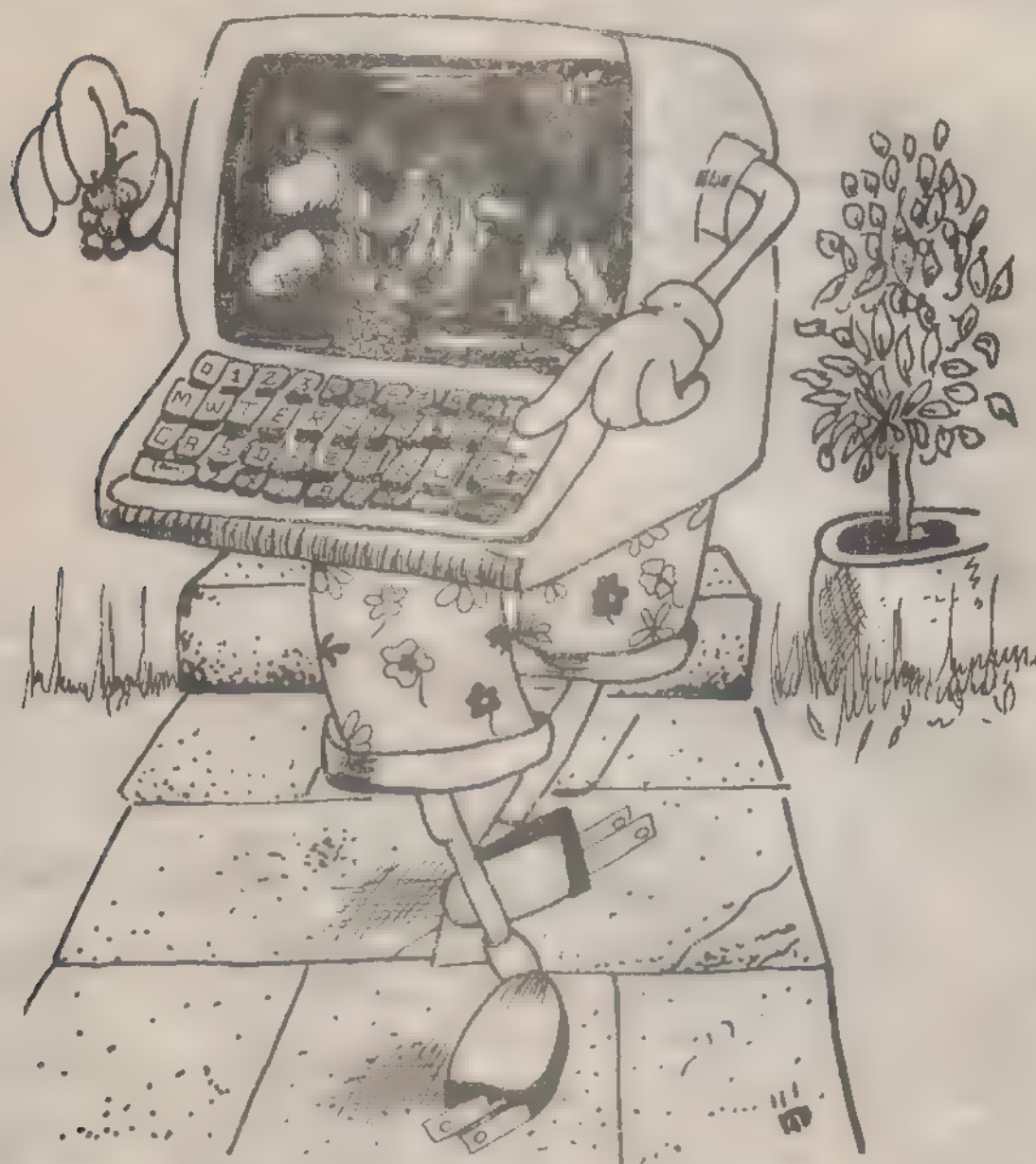
La ultima alternativa:

10 PRINT A\$(1 TO) cuesta 18 bytes tal como Ud. se lo imaginaba

10 PRINT A\$(A TO B) cuesta 13 bytes

comparados con los 25 de la linea que usa directamente valores.

- 22 -



Maquina de frutas.

```

00000 LET A=0
00001 PRINT "HELLO WORLD?"
00002 INPUT T
00003 IF T#="" THEN
00004   FOR X=1 TO 3
00005     LET N#="+*P+-.*"(INT (RN
00006       +1)+1)
00007     NEXT X
00008     GO TO 1
00009   FOR X=1 TO 5
00010     IF INKEY#"" THEN GOTO 200
00011   NEXT X
00012   GOTO 100
00013   IF INKEY#"" THEN GOTO 100
00014   LET U=-10
00015   IF Z$(1)= "+" THEN LET U=2
00016   IF Z$(1)=Z$(3) OR Z$(3)=Z$(
00017     ) OR Z$(1)=Z$(5) THEN LET U=5
00018   IF Z$(1)=Z$(3) AND Z$(1)="+"
00019     THEN LET U=10
00020   IF Z$(1)=Z$(3) AND Z$(1)=Z$
00021     THEN LET U=50
00022   PRINT "PAGE3";U
00023   SCROLL
00024   LET S=S+U
00025   GOTO 100
00026   PRINT "UD. DBTUVO";S

```



```

1 REM 16K MAQUINA DE AJAJA
2 REM *****
30 PRINT "COMIENZO VELOCIDAD"
40 IF F<1 THEN GOTO 20
40 PRINT "CUANTO TIEMPO QUIERE"
50 INPUT N
60 IF N<1 OR N>10 THEN GOTO 40
70 POKE 16437,188+N+12
80 IF PEEK 16437=128 THEN GOTO 1000
1000 LET F=F-(N/10)
1100 IF F<1 THEN LET F=1
1200 STOP
2000 SCROLL
3000 PRINT "***TIEMPO ARAJAJA**"
4000 SCROLL
5000 GOTO 1000

```

El programa le mostrara continuamente una seleccion de tres caracteres, y Ud. podra presionar una tecla cuando sea impresa una combinacion ganadora. Si es lo suficientemente rapido su puntaje se incrementara rapidamente, ya que el puntaje depende de cuantas combinaciones haya sido capaz de parar. Las combinaciones ganadoras son:

*** paga 2

*** paga 10

cualquiera que tenga 2 caracteres iguales paga 5 (excepto ***)

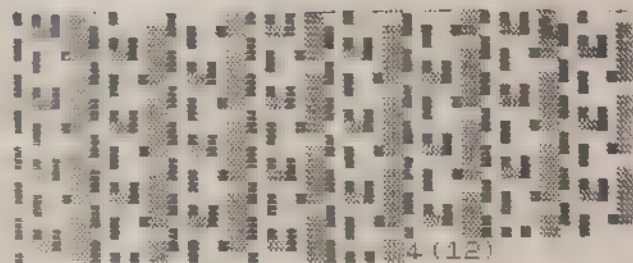
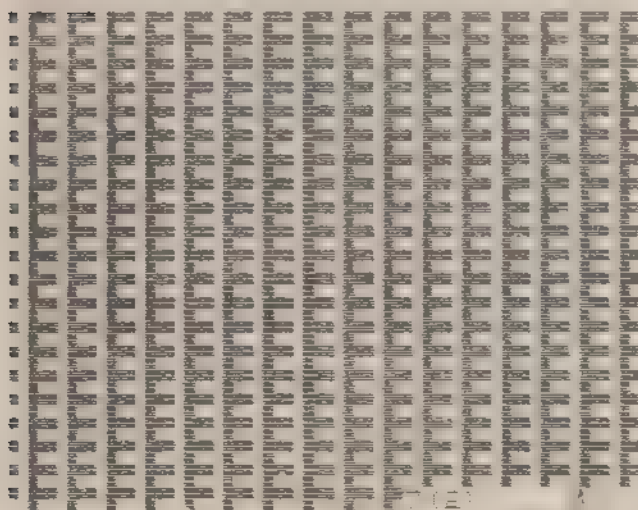
Cualquiera que tenga 3 caracteres paga 50

Inicialmente puede elegir la velocidad para seleccionar. Presione "Q" para hacer su score. Si Ud. presiona "Q" cuando esta seleccionando perdera 10 puntos.

16 K: si posee expansion a 16 K agregue este programa.

Patrones Universales

Este programa produce patrones graficos al azar. Hay mas de dos millones de disenos que pueden verse en pantalla o imprimirse. Cada patron tiene un numero y una longitud de string impresas en el fondo. Con esos datos puede ser llamado nuevamente. Responda que NO cuando le pregunta por patrones al azar y entre esos numeros.



```

10 REM A(33)
20 PRINT "MODELO AL AZAR? (S/N)
30 INPUT A$
40 CLS
50 IF NOT A$="S" THEN GOTO 200
60 LET L=INT (RAND*33)+1
70 LET X=INT (RAND*65536)+1
80 RAND X
90 FIRST
100 FOR J=1 TO L
110 LET A(J)=INT (RAND*10)+1
120 NEXT J
130 FOR K=1 TO 65536/L
140 FOR R=1 TO L
150 PRINT CHR$( A(K)),
160 NEXT K
170 NEXT J
180 PRINT X;"("L)";"
190 STOP
200 PRINT "NUMERO MODELO?"
210 INPUT X
220 CLS
230 PRINT "LONGITUD DEL CORDON"
240 INPUT L
250 CLS
260 GOTO 80

```

CARRERA

```

1 GO SUB 1000 LET s=2: BORDER
2 PAPER 1: CLS
3 PRINT AT 0,0: INK 8:

```

LA CARRERA

```

5 IF s=3 THEN BEEP .3,12: BEE
P .9,6: BEEP .3,15: GO TO 10
6 LET a=a+1: FOR f=1 TO 4: LE
T 0=.0123: BEEP 0,2: BEEP 0,4: B
EEP 0,6: BEEP 0,8: BEEP 0,10: BEE
EP 0,12: BEEP 0,14: BEEP 0,16: BEE
EP 0,2: BEEP 0,4: BEEP 0,6: NE
T 10 GO TO 3

```

```

10 INK 7: CLS LET s=0: LET t
1000=0
15 PRINT AT 0,15: "CARRERA"
16 PRINT AT 0,15: "(A)--IZQ.", "(S)--DER.", "(X)--
ARIBA", "(M)--ABAJO"
17 PRINT AT 0,15: "ES UNA CARRERA CO
TRA EL RELOJ"
20 PRINT AT 18,2: "PRESIONE ALG
UNA TECLA PARA EMPE
ZAR": PAUSE 8

```

```

51 REM *****
52 REM * "A" en lineas *
53 REM * 210,240 y 280 de *
54 REM * ben escribiase en *
55 REM * MODO GRAFICO *
56 REM *****
90 LET a=1: LET b=100: LET t

```

```

P=1: BORDER 1: PAPER 1: INK 8: C
LS

```

```

100 FOR f=0 TO 31: PRINT AT 1,
PAPER 2: "*" AT 20,f: "+" NEXT
f

```

```

101 FOR f=1 TO 20: PRINT AT 2,
PAPER 2: "*" AT 10,1: "*" AT 10,2:
NEXT f

```

```

102 FOR f=3 TO 18: PRINT AT 3,
PAPER 5: "*" AT 13,3: "*" AT 13,4:
5: "*" AT 13,5: "*" AT 13,6: "*" AT 13,7:
NEXT f

```

```

103 FOR f=2 TO 18: PRINT AT 4,
PAPER 6: "*" AT 16,2: "*" AT 16,3:
5: "*" AT 16,4: "*" AT 16,5: "*" AT 16,6:
NEXT f

```

```

104 FOR f=7 TO 10: PRINT AT 5,
PAPER 3: "*" AT 12,7: "*" AT 12,8:
7: "*" AT 12,9: "*" AT 12,10:
NEXT f

```

```

105 FOR f=9 TO 12: PRINT AT 6,
PAPER 3: "*" AT 9,9: "*" AT 9,10:
10: "*" AT 9,11: "*" AT 9,12:
NEXT f

```

```

106 FOR f=20 TO 22: PRINT AT 3,
PAPER 4: "*" AT 9,3: "*" AT 15,3:
3: "*" AT 15,4: "*" AT 15,5:
NEXT f

```

```

107 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

108 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

109 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

110 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

111 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

112 FOR f=22 TO 24: PRINT AT 6,
PAPER 4: "*" AT 12,6: "*" AT 15,6:
6: "*" AT 15,7: "*" AT 15,8:
NEXT f

```

```

120 FOR f=3 TO 15 STEP 6: PRINT
AT 7,8: PAPER 3: "*" AT 7,23: PA
PER 4: "*" NEXT f

```

```

121 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

122 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

123 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

124 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

125 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

126 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

127 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

128 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

129 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

130 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

131 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

132 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

133 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

134 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

135 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

136 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

137 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

138 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

139 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

140 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

141 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

142 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

143 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

144 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

145 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

146 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

147 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

148 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```

```

149 FOR f=6 TO 18 STEP 6: PRINT
AT 7,11: PAPER 3: "*" AT 7,21: P
APER 4: "*" NEXT f

```


CUENTAS PERSONALES

Este programa, el cual llevara todas las cuentas de su casa, le ahorrara tiempo, pero ese no es el beneficio mas importante que este paquete ofrece. La siguiente lista de trabajos puede ser codificada en el programa y cargarla le llevara solo unos segundos.

TRABAJO QUE SERAN COMPUTADOS

- 1- Archivos con detalle de los pagos regulares y los datos de cuando caera ■ vencimiento.
- 2- Los cheques que seran deducidos proxiamente.
- 3- Recordar los proximos pagos y ver que no se haya olvidado de agregar alguno.
- 4- Decidir cuando sus cheques por salario seran agregados, y costos fijos deducidos.
- 5- Permitir la posibilidad de pagos especiales dentro y fuera de las cuentas ya establecidas.
- 6- Hacer convenios para cambiar los costos fijos.

Para hacer siempre el trabajo, el programa deduce los costos fijos el mismo dia del mes en que acredita el salario. Se ha usado como dia de pago el 25 de cada mes.

La maquina decide cuando preguntar por el monto del cheque de pago basandose en dos datos, buscados en la memoria.

Los datos entrados antes de dar el RUN al programa son recordados por la maquina y los nuevos datos se los preguntara ni bien Ud.haya cargado el programa. La primera parte del programa es un sistema para el manejo de caja. Puede ser cargado desde el cassette y podra usarlo como base de este programa.

La seccion siguiente al manejo de caja es el menu principal, el que aparecera en pantalla tan pronto como Ud. termine de cargar el programa.

```

1000 GOSUB 200
1010 PRINT "CUENTAS PERSONALES"
1020 PRINT "-----"
1030 PRINT "BALANCE HASTA LA FECH"
1040 PRINT "ARRIETE"
1050 PRINT "EXPOSICION DEL SISTEMA"
1060 PRINT "ARRIETE"
1070 PRINT "MODIFICAR LOS COSTOS F"
1080 PRINT "ARRIETE"
1090 PRINT "ARCHIVAR"
1100 PRINT "ARRIETE"
1110 GOTO 200
2070 INPUT A$
2080 IF A$="C" THEN GOTO 2000
2090 IF A$="S" THEN GOTO 3000
2100 IF A$="O" THEN GOTO 4000
2110 IF A$="T" THEN GOTO 5000
2120 GOTO 200

```

La linea 290 hace que se retorne al menu, pues la seleccion hecha no esta contenplada ■ el menu. La proxima seccion d

La proxima seccion de codigos es opcional y es muy larga. Es factible que la primera vez que Ud. corra el programa, ella elija por si sola, pero eso no sucedera de nuevo, y le proporcionara un buen espacio. Una alternativa y un metodo mucho mas simple para elegir el sistema dado despues de listarlo pero ello demanda el uso directo de comandos y la inexperiencia de no usarlo sera facilmente superada.

Si Ud. decide no incluir la seccion del programa de exposicion del sistema, lo mismo podra encontrar el resto del listado mas claro que si lo hubiera hecho a traves de el.

En el texto son incluidos algunos detalles sobre almacenamiento de datos y cobranzas.


```

300CLS
310 LET BL=0
320 LET D$="MES VENCIDO"
340 PRINT "ESTA EN CREDITO O PA
3400 DE CARGO CON EL BANCO? C/O"
350 INPUT B$
360 GOSUB 10
370 IF B$="0" THEN GOSUB 30
380 IF B$="0" THEN GOSUB 35
390 IF B$="0" AND B$="0" THEN
GOTO 300

```

Este programa permite llevar cuenta de hasta 10 costos fijos para ser pagados por mes, 10 que son pagados trimestralmente, y 10 que sean pagados una vez al año.

Los datos seran cargados en forma de vectores que seran definidos de la siguiente forma:

```

400DIM B(10)
410 DIM N(10)
420 DIM M(10)

```

Para cargar los elementos de los vectores se usan dos loops. Un loop se maneja con la letra "K", y el otro es controlado con la letra "J".

Las siguientes son las lineas en BASIC que hacen ese trabajo.

```

430FOR K=1 TO 3
440 FOR J=1 TO 10
450 IF K=1 THEN PRINT "MENSUALM
ENTE";
460 IF K=2 THEN PRINT "CUATRIME
STRALMENTE";
470 IF K=3 THEN PRINT "ANUALMEN
TE";
480 PRINT "*NO HAY ORDEN FIJA*"
J
490 GOSUB 10
500 IF J=3 THEN NEXT K

```

Esta seccion imprime las respuestas.

La maquina le preguntara por el monto de cada costo fijo, uno por vez. Si Ud. responde con cero, entonces asumira que todos los costos fijos del mes han sido entrados antes y pasara a pedirle los costos fijos trimestrales.

Si los 10 costos fijos trimestrales han sido entrados la maquina automaticamente pasara a pedirle los costos fijos anuales.

La maquina usa los valores cargados en "K" para decidir que es lo que hace.

```

510IF K=2 THEN GOTO 590
520 IF K=3 THEN GOTO 690
530 IF K=4 THEN GOTO 800
540 GOSUB 70
570 NEXT J
580 NEXT K

```

Si "K" no es igual a dos o tres o aun a cuatro, entonces elegira uno de los montos de pago que fue cargado en el vector M elemento J. El arreglo esta cargado con informacion sobre pagos mensuales.

```

590PRINT D$
600 INPUT A
610 PRINT TAB(8); "MENS*"A
620 IF A$="0" THEN GOTO 590

```

El proximo trabajo es cargar el monto de los pagos trimestrales y el mes durante el cual deberan ser pagados; ambos son almacenados en la misma variable.

Supongamos un monto regular de \$a 999.99, entonces si a este monto lo dividimos por mil, el resultado sera menor que uno. Asi si Ud. debe pagar \$a 103.75 el valor almacenado sera 0.10375.

Si entonces el numero del mes para el proximo pago es agregado al elemento, para el mes de Diciembre el valor que mantiene almacenado sera 12.1035. Para saber el mes se toma el valor entero del numero; para el monto un poco mas complicado.

1) Toma el valor del entero y lo multiplica por 100000 y llama a esto A. 2) Multiplica el valor en el elemento por 100000 y lo llama B. 3) Resta: A-B y divide por 100.

Veamoslo:

- 1) $12 \times 100000 = 1200000$A=1200000
- 2) $1210375 \times 100000 = 1210375$B=1210375
- 3) A-B=10375

Esto parece un poco largo pero en realidad ahorra espacio, y es un buen metodo para extraer informacion.

```
650 LET Q(U)=A/100000
660 NEXT U
670 NEXT K
```

Es decir si el mes y el monto de pago fueran almacenados en variables separadas tendríamos al menos 100 bytes extras para el almacenamiento.

Otra forma del programa que sigue a continuacion, guarda los valores en un vector (N) que mantiene las ordenes anuales.

```
680 PRINT C$
710 INPUT A
730 LET N(U)=A/100000+A
740 NEXT U
750 NEXT K
```

Al fin del loop controlado por "K" el contador debe ponerse en 4 y luego el loop no se inicia. La computadora reconoce esto como una senal en la linea 530 y salta a la linea 800 donde esta la rutina que muestra el estado de "costos fijos".

```
800 CLS
810 GOSUB 60
820 GOSUB 70
830 IF A$="C" THEN GOTO 400
840 CLS
```

Ir a 480, en la linea 630 tiene el efecto de redimensionar los vectores con la misma variable. Habiendo hablado ya de la linea 80 es hora que demos su listado.

```
800 PRINT "ORDENES FIJOS"
810 LET Z=100000
820 PRINT "NO." TAB 24; "MES"
830 PRINT "13." "E TRIMESTRE" TAB 24
840 PRINT "E ANUAL"
850 FOR J=1 TO 10
860 PRINT J; TAB 4; M(J); TAB 1
870 PRINT (INT (Q(J)*Z+.5)-INT Q(J)*Z)
880 PRINT (TAB 20; (INT Q(J) * TAB 24
890 PRINT (INT (N(J)+Z+.5)-INT N(J)*Z)/10
900 PRINT TAB 32; INT N
910 NEXT J
920 RETURN
```

La linea 84 es compleja pero hubiera sido peor de no ser por la linea 81 que evita el manejo de los "100000". La explicacion ya hecha del almacenamiento y el metodo de extraccion sirve para entender esta linea.

La linea 900 inicia otro segmento que se basa en otras tres variables para mantener otro dato.

Este es el dato que debe recordarse desde la primera ejecucion (RUN) a la proxima y es el cual le ayudara a decidir cual pago se debe desde la ultima vez que fue usado.

```
900 GOSUB 60
910 LET D4=D1
920 LET D5=D3
930 CLS
```


Luego sigue la parte que carga en detalle de los libros de cheque y "libros de pagos".

```

1000CLS
1010 PRINT "TECLEE EL NUMERO DE
SU CHEQUERA"
1020 INPUT C
1030 PRINT C
1040 PRINT "AHORA EL SIGUIENTE NR
O, DE CHEQUE"
1050 INPUT C1
1070 PRINT "PROXIMAMENTE EL ULTI
MO NR.O. DE CHEQUE EN LA CHEQUERA"
1080 INPUT C2
1090 PRINT C2
1100 PRINT "Y EL NR.O. DE PROXIMO
PAGO ATRASADO"
1110 INPUT C3
1120 PRINT C3
1130 GOSUB 70
1140 IF A$="C" THEN GOTO 1000
1150 GOTO 200

```

Este es el fin de la rutina de "puesta a punto", una vez usada puede borrarse.

Como ya fue mencionado los "costos fijos" pueden no mantenerse fijos en el mismo nivel para siempre. Para ello hay una rutina que permite dichos arreglos. Las líneas desde 1200 hacen uso de la subrutina de la línea 80 para imprimir el mapa de "costos fijos".

```

1200CLS
1210 PRINT "REVISADO"
1220 GOSUB 80
1230 PRINT
1240 PRINT "QUE ORDEN FIJA?"
1250 INPUT J
1260 PRINT J
1270 PRINT "MES(1) CUATRIMESTRE"
1280 PRINT "2" OR ANIO(1310"
1290 INPUT A
1300 CLS
1310 PRINT "ENTREGAR VALOR CORRE
CTO"
1320 GOSUB 10

```

Ahora la computadora tiene toda la información para seleccionar el correcto elemento y guardar el dato en el correspondiente vector. Las dos líneas siguientes seleccionan la apropiada rutina de carga.

```

1330IF A=2 THEN GOTO 1380
1340 IF A=3 THEN GOTO 1430

```

Si A es otra cosa, esta debe ser para un costo fijo mensual.

```

1340LET M(J)=AJ
1350 GOTO 200
1360 PRINT D$
1370 INPUT A
1400 LET Q(J)=AJ/1000+A
1410 GOTO 200
1420 PRINT D$
1430 INPUT A
1450 LET N(J)=AJ/1000+A
1460 GOTO 200

```

Ahora sigamos con un negocio regular.

```

2000CLS
2010 PRINT "TRANSACCIONES DESDE"
2020 PRINT "D4, "/" D5:" "/" D6
2030 GOSUB 60
2040 IF (D3-D5)*365+(D2-D5)*30+D
1-D4<0 THEN GOTO 2030

```

Esta ultima linea prevee los cambios por atraso en tiempo. Todo tipo de dificultades pueden surgir. Los salarios pueden ser pagados una vez y contados dos. Si el interes debe ser calculado, entonces el nuevo dato no puede ser alcanzado por el loop que genera al interes ya que se estaria en un ciclo cerrado. La linea calcula el numero de dias que ha pasado y se niega a continuar si dicho numero es negativo.

La proxima seccion es un poco es compleja de escribir.

```
2050 PRINT D1;"/",D2;"/";D3
2060 IF D5=D2 AND D6=D3 THEN GOT
O 2180
2070 IF D4<25 THEN GOSUB 2130
2080 LET D4=1
2090 LET D5=D5+1
2100 IF D5>12 THEN LET D6=D6+1
2110 IF D5>12 THEN LET D5=D5-12
2120 GOTO 2060
2130 PRINT "TECLEE SALARIO POR M
ES:" D5;"/";D6
2140 GOSUB 10
2150 GOSUB 35
```

Habiendo decidido cual mes de pago a cargar, la computadora continua decidiendo cual de los "costos fijos" se deben.

Estamos en el medio de la rutina de busqueda que se inicia en la linea 2130. Esta rutina llama a otras tres, las cuales no han sido listadas todavia. La rutina de la linea 90 es la que decide sobre "costos fijos".

```
2160 GOSUB 90
2170 RETURN
2180 IF D4>24 OR D1<25 THEN GOTO
2200
2190 GOSUB 2130
2200 FOR J=1 TO 10
2210 LET A=M(J)
2220 GOSUB 30
2230 NEXT J
2240 FOR J=1 TO 10
2250 GOTO 120
2260 LET A=(M(J)-INT M(J))*1000
2270 GOSUB 30
2280 NEXT J
2290 FOR J=1 TO 10
2300 IF INT N(J)>D5 THEN GOTO 1
04
2310 LET A=(N(J)-INT N(J))*1000
2320 GOSUB 30
2330 NEXT J
2340 RETURN
```

Ud. debe recordar que unicamente un mes del trimestre en el que debe ser pagado el "costo fijo" puede ser cargado al arreglo.

La seccion del programa que comienza en la linea 120 usa ese mes para chequear el numero de mes si alguno de los otros tres no han aparecido. Si el presente numero de mes, cargado en la maquina, es alguno de los cuatro meses el que debe ser hecho el pago, entonces el programa va a la linea 97 y deduce el pago para el balance. En cualquier otro mes, esta seccion es saltada y el control es enviado a la linea 99.

Los pagos anuales son faciles. Ellos deben ser pagados solo cuando el "INT N(J)" (la parte entera del valor del elemento J del vector N), es igual al numero de mes presente.

```
100 LET A=INT C
110 IF D5=A OR D5=A-3 OR D5=A-6
OR D5=A+9 OR D5=A-3 OR D5=A-6 O
R D5=A-9 THEN GOTO 97
140 LET A=0
150 GOTO 99
```

El resto del listado es muy simple, comparado con el resto.

Primero, todos los cheques que han sido escritos desde la ultima vez que el balance fue hecho, seran deducidos. Si un valor 0 es entrado, la computadora preguntara si el cheque fue cancelado; si la respuesta es "NO" entonces ella tomara el "INPUT" como senal de que el ultimo cheque ha sido destruido, y preguntara por los pagos con atraso.

```
2200 CLS
2210 GOSUB 45
2220 PRINT "NRO. DE CHEQUERA:";C
2230 PRINT "TECLEE EL VALOR DEL
NRO. DE CHEQUE:";C1
2240 GOSUB 10
2250 GOSUB 30
```



```

2300 IF A0=0 THEN GOTO 2450
2310 IF C1=C2 THEN PRINT "SI UD.
      TIENE DETALLES DE LA CHEQUERA"
2320 IF C1=C2 THEN PRINT C+1 "CL
2330 TECLÉE D.DE OTRA FORMA TECL
      ENTER"
2340 IF C1=C2 THEN INPUT A$
2350 IF C1=C2 AND A$="D" THEN GO
      TO 2340
2360 IF C1=C2 THEN GOTO 2470

```

Todo esto necesita una o dos palabras de explicación. Si C1 es igual a C2 entonces el próximo cheque es el último en la chequera. Si la próxima chequera ya le llegó del banco, entonces los datos podrán ser cargados después de pulsar la "D". Si, de otra forma, el banco se demora en enviarle la nueva chequera, Ud. no podrá escribir nada y el programa seguirá con su flujo.

```

2370 LET C1=C1+1
2380 GOTO 2300
2390 LET C=C+1
2400 CLS
2410 PRINT "TECLÉE EL PRIMER NRO
      DE CHEQUE EN LA CHEQUERA "+"C
2420 INPUT C1
2430 PRINT C1
2440 PRINT "AHORA EL ULTIMO"
2450 INPUT C2
2460 PRINT C2
2470 GOSUB 70
2480 IF A$<>"C" THEN GOTO 2350
2490 GOTO 2500

```

Ahora viene la rutina de cancelación de cheques:

```

2450 PRINT "SI ESE FUE UN CHEQUE
      CANCELADO TECLÉE C.DE OTRA FORM
      A TECLÉE ENTER"
2460 INPUT A$
2470 IF A$="C" THEN LET C1=C1+1
2480 CLS
2490 IF A$="C" THEN GOTO 2200

```

Ahora que todas las posibilidades ya han sido ingresadas, comenzamos el balance con la línea 2490.

```

2490 GOSUB 45
2500 PRINT "PAGOS EN CTA. CORRIE
      NTE"
2510 PRINT
2520 PRINT "TECLÉE VALOR DEL NRO
      DE PAGO ATRASADO+" C3
2530 GOSUB 10
2540 IF A0=0 THEN GOTO 2600
2550 GOSUB 35
2560 LET C3=C3+1
2570 GOTO 2500

```

Y ahora los pagos especiales que le hagan a que Ud. haya hecho sobre montos que no han sido trabajados con cheques.

```

2600 CLS
2610 GOSUB 45
2620 PRINT "ALGUN DESEMBOLOSO ESP
      ECIAL? S/N"
2630 INPUT A$
2640 IF A$<>"Y" THEN GOTO 2690
2650 GOSUB 10
2660 GOSUB 30
2670 GOTO 2600
2680 CLS
2690 PRINT "ALGUN DEPOSITO ESPEC
      IAL? S/N"
2700 INPUT A$
2710 IF A$<>"Y" THEN GOTO 200
2720 GOSUB 10
2730 GOSUB 35
2740 GOTO 2690

```

Finalmente, la rutina de comienzo automatico sera impresa desuaves del balance general).

```

0000 CLS
0010 PRINT TAB 8; "BALANCE ESPECI
0020 PRINT
0030 GOSUB 40
0040 PRINT AT 10,10; "PREPARE EL
0050 PRINT TAB 10; "TECLEE ENTER"
0060 INPUT A$
0070 LET D4=D1
0080 LET D5=D2
0090 LET D6=D3
0100 CLS
0110 SAVE "CAJA"
0120 GOTO 200

```

Si, en cambio, Ud. ha omitido la rutina de inicio, hay tro trabajo por hacer antes de que el programa sea corrido sin que aborte.

PROCEDIMIENTO PARA EL COMIENZO

Antes que todo, deletee las lineas 240 a la 282 de la seccion del menu y entonces teclee estos comandos sin numero de linea. La computadora ejecutara en forma correcta y la informacion sera recordada desde un "RUN" a otro.

Mire todo el listado para estar seguro ue toda la informacion ha sido entrada antes de comenzar.

- 1) LET BL=(balance de su banco)/ENTER.
- 2) DIM H(10)/ENTER.
- 3) DIM N(10).
- 4) DIM Q(19)/ENTER.
- 5) LET D4=(numero de dia)/ENTER.
- 6) LET D5=(numero de mes)/ENTER.
- 7) LET D6=(los dos ultimos digitos del ano)/ENTER.

Teclee 60 TO 200 y el menu principal aparecera. Presione "C" para cambiar los costos fijos y luego cambie por cero y mes cero los montos pagables y el mes en que vence el pago.

Los pagos trimestrales necesitan solo un numero de mes, no cuatro.

Su programa esta listo para correr.

VARIABLES USADAS

Para el manejo de caja:

- AJ ajuste de caja
- A\$ para seleccionar
- BL balance
- D1 numero de dia
- D2 numero de mes
- D3 numero de año (solo los dos ultimos digitos)

Variables especificas:

- W variable de seleccion para ser usada cuando A\$ permanezca fijo
- A cualquier input que no sea permanente
- D\$ mes de vencimiento
- C numero de chequera
- C1 numero del proximo cheque
- C2 ultimo cheque en la chequera
- C3 numero del proximo pago atrasado
- D4 a D6 para cargar datos secundarios
- J y K contadores
- Z ajusta automaticamente a 100000
- Q, N y M arreglos dimensionados en 10 para cargar los costos fijos

15 Pulsos

```

3  DIM B(16)
10  GOSUB 300
15  LET M=0
20  FOR I=1 TO 16
30  LET A(I)=0
40  NEXT I
50  FOR I=1 TO 16
60  LET R=RND*16
70  IF NOT A(R)=0 THEN GOTO 60
80  LET A(R)=I
90  NEXT I
100 GOSUB 500
110 IF F=1 THEN GOTO 20
120 GOSUB 600
130 PRINT
140 PRINT "SU JUGADA"
150 INPUT X
160 LET C=X
170 IF X=0 THEN STOP
180 GOSUB 400
190 GOSUB 700
200 IF NOT F=0 THEN GOTO 210
210 PRINT
220 PRINT C;"JUGADA ILEGAL. VUELVA A JUGAR"
230 GOTO 150
240 LET A(X+F)=A(X)
250 LET A(X)=16
260 GOTO 800
270 LET M=M+1
280 GOTO 120
290 LET B(1)=2
300 LET B(2)=4
310 LET B(3)=5
320 LET B(4)=7
330 LET B(5)=10
340 LET B(6)=12
350 LET B(7)=13
360 LET B(8)=15
370 RETURN
380 REM CONVERTIR EL NRO EN UN LUGAR DEL ARREGLO
390 FOR I=1 TO 16
400 IF A(I)=X THEN GOTO 430
410 NEXT I
420 LET X=X-I
430 RETURN
440 REM VERIFICA SI ES UNA SOLUCION POSIBLE
450 LET F=1
510 LET F=0
520 FOR I=1 TO 16
530 FOR J=I+1 TO 16
540 IF A(I)>A(J) THEN LET F=F+1
550 NEXT J
560 NEXT I
570 FOR I=1 TO 8
580 IF A(B(I))=16 THEN LET F=F+1
590 NEXT I
600 IF (3/2)*2=3 THEN LET F=0
610 RETURN
620 REM MOSTRAR TABLA DE JUEGOS
630 CLS
640 PRINT "QUINCE PULSOS"
650 PRINT
660 LET I=1
670 PRINT
680 FOR Y=1 TO 4
690 IF A(I)<10 THEN PRINT " "
700 IF A(I)=16 THEN PRINT " "
710 IF NOT A(I)=16 THEN PRINT A(I)
720 PRINT " "
730 LET I=I+1
740 NEXT Y
750 PRINT
760 PRINT
770 IF I=17 THEN RETURN
780 GOTO 625
790 REM PROBAR PARA JUGADA LEGAL
800 LET F=0
810 IF X+1>16 THEN GOTO 725
820 IF A(X+1)=16 THEN LET F=1
830 IF X-1<0 OR X-1=0 THEN GOTO 725
840 IF A(X-1)=16 THEN LET F=-1
850 IF X+4=16 THEN GOTO 745
860 IF A(X+4)=16 THEN LET F=4
870 IF X-4<0 OR X-4=0 THEN GOTO 725
880 IF A(X-4)=16 THEN LET F=-4
890 RETURN
900 REM CHEQUEANDO EL GANADOR
910 FOR I=1 TO 16
920 IF NOT A(I)=I THEN GOTO 940
930 NEXT I
940 GOSUB 600
950 PRINT
960 PRINT
970 PRINT "LE LLEVO SOLO";M;"MOVIDAS"
980 STOP

```

Este programa es de un juego que fue inventado por Sam Loyd en 1878 y que consiste de una matriz de 4x4 que tiene 15 bloques sin numerar. El objeto del juego es terminar con todos los bloques numerados. Suena fácil pero no lo es.

No todas las combinaciones de comienzo factible son: 20, 922, 789, 888, y 000, así que el juego las chequea. Si no su jugada de comienzo la computadora blanquea el display antes que Ud. pulse. Es por ello que antes de mostrar cualquier cosa en pantalla se demora algunos segundos. El programa también chequea por movidas ilegales en cada entrada.

Con cada entrada, la computadora chequea para ver si gana. Si Ud. lo hace ella lo felicitará y le dirá cuantas jugadas le tomó hacerlo.

Si logra frustrarlo, pulse 0 (cero) y el juego termina.

Solitario

```

10 REM "SOLITARIO"
20 DATA 0,54,127,127,62,28,6,2
30 DATA 0,8,28,62,127,62,28,6
40 DATA 0,8,28,42,127,42,127,4
2,8,28
50 DATA 0,8,28,62,127,42,8,28
60 DATA 0,70,201,73,73,73,230
70 DATA 153,90,60,255,255,60,9
J,150
80 FOR i=144 TO 149: FOR j=0 TO
0 7
90 READ a: POKE USA CHR$ i+j,a
100 NEXT j: NEXT i
110 BORDER 4: PAPER 4: INK 0: C
LS
120 DIM a(6,7): DIM b(17): LET
b(1)=17: LET b$=""
130 LET a$="": FOR i=1 TO 52: L
ET a$=a$+CHR$ i: NEXT i
140 PRINT AT 11,11;"BARAJANDO"
150 FOR i=1 TO 52
160 LET w=INT (RND*52+1)
170 IF CODE a$(w) <> 0 THEN GO TO
200
180 LET w=w+1: IF w=53 THEN LET
w=1
190 GO TO 170
200 LET b$=b$+CHR$ w
210 LET a$(w)=CHR$ 0
220 NEXT i
230 LET a=0
240 FOR i=1 TO 7
250 FOR j=2 TO 6
260 LET a=a+1
270 LET a(j,i)=CODE b$(a)
280 NEXT j: NEXT i
290 FOR i=1 TO 7: LET a(1,i)=6:
NEXT i
300 LET a=a+1: LET b=CODE b$(a)
310 FOR i=2 TO 17: LET a=a+1: L
ET b(i)=CODE b$(a): NEXT i
320 CLS: PRINT " " FOR i=1
TO 7: PRINT STR$ i: " " NEXT
i: PRINT
330 FOR i=2 TO 5: PRINT
340 FOR j=1 TO 7: LET a=a(1,j):
GO SUB 800
350 PRINT " " PAPER 7:b$: " "
360 NEXT j
370 PRINT
380 FOR j=1 TO 7: LET a=a(1,j):
GO SUB 800
390 PRINT " " PAPER 7:a$: " "
400 NEXT j
410 PRINT: FOR j=1 TO 7: PRINT
" " PAPER 7:" " NEXT j
420 NEXT i
430 PRINT: FOR j=1 TO 7: LET a
=a(6,j): GO SUB 800: PRINT " "
PAPER 7:b$: " " NEXT j
440 PRINT: FOR j=1 TO 7: LET a
=a(6,j): GO SUB 800: PRINT " "
PAPER 7:a$: " " NEXT j

```

	1	2	3	4	5	6	7	
1	♥	♥	♠	♠	♠	♠	♠	♠
2	♥	♥	10	6	♥	4	♥	♠
3	♥	♥	♠	♠	♠	♠	♠	♠
4	♥	♥	♠	♠	♠	♠	♠	♠
5	♥	♥	♠	♠	♠	♠	♠	♠
6	♥	♥	♠	♠	♠	♠	♠	♠
7	♥	♥	♠	♠	♠	♠	♠	♠
8	♥	♥	♠	♠	♠	♠	♠	♠
9	♥	♥	♠	♠	♠	♠	♠	♠
10	♥	♥	♠	♠	♠	♠	♠	♠
11	♥	♥	♠	♠	♠	♠	♠	♠
12	♥	♥	♠	♠	♠	♠	♠	♠
13	♥	♥	♠	♠	♠	♠	♠	♠
14	♥	♥	♠	♠	♠	♠	♠	♠
15	♥	♥	♠	♠	♠	♠	♠	♠
16	♥	♥	♠	♠	♠	♠	♠	♠
17	♥	♥	♠	♠	♠	♠	♠	♠

Cartas en el monton: 16

La idea es sacar todas las cartas de la pila del costado. Una carta se puede sacar si es la anterior o la posterior a la carta que esta en la pila. Ejemplo: si en la pila hay un 4 se puede sacar de las columnas un 3 o un 5. De cada columna solo se puede sacar la ultima carta. Para hacerlo presione la tecla correspondiente al numero de columna de la que quiere sacar. Si no tiene ninguna posibilidad de sacar cartas debera pedir una del monton presionando el 0 (cero). El juego termina cuando logro dejar vacias todas las columnas de cartas o bien porque ya no le quedan mas cartas en el monton (tiene 16 cartas para todo el juego). Buena suerte!

NOTA GRAFICA: los graficos se obtienen de:

GRAPHICS "A" corazon
 GRAPHICS "B" diamante
 GRAPHICS "C" trebol
 GRAPHICS "D" pica



```

450 PRINT FOR I=1 TO 7 PRINT
460 PAPER 7: NEXT I
470 PRINT FOR J=1 TO 7 LET a
=8+J: GO SUB 800 PRINT
PAPER 7: "a$": NEXT J
470 PRINT FOR I=1 TO 7 LET a
=8+I: GO SUB 800 PRINT
PAPER 7: "a$": NEXT I
480 PRINT AT 21,22 INK 0 "Cartas
=800 GO SUB 800 10 PAPER 7
490 GO SUB 800 GO SUB 890
500 LET a$=INKEY$
510 IF a$<"0" OR a$>"7" THEN GO
TO 510
520 LET c=VAL a$. IF NOT c THEN
GO TO 730
530 IF a(1,c)=1 THEN BEEP 1,0
GO TO 510
540 LET a=a(a(1,c),c) GO SUB 8
00 LET d=w
550 LET a=b GO SUB 800
560 IF d=13 AND w=1 THEN GO TO
800
570 IF d=1 AND w=13 THEN GO TO
600
580 IF ABS (d-w)>1 THEN BEEP 1
,0 GO TO 510
590 BEEP .25,30
600 LET b=a(a(1,c),c) GO SUB 8
00
610 LET a(1,c)=a(1,c)-1 LET d=
(a(1,c)-1)*3+1
620 LET a=a(a(1,c),c) GO SUB 8
00 LET c=(c-1)*4+1
630 IF d=1 THEN FOR I=2 TO 6 P
RINT AT 1,c) PAPER 4: " " NEXT
I GO TO 670
640 PRINT AT d,c) "AT d+1,c)
" "a$": AT d+2,c) "b$
650 PAPER 4: PRINT AT d+3,c)
"AT d+4,c) " "AT d+5,c)
PAPER 7
660 FOR I=1 TO 7 IF a(1,I)=1
THEN GO TO 510
670 NEXT I PAPER 4 INK 0 PRI
NT AT 11,3 "FELICIDADES LO CONSE
GISTE!"
680 FOR d=1 TO 20
690 SOUND 6,15,7,7,8,16,9,16,10
,12,16,13,8
700 PAUSE 0
710 NEXT d
720 PAPER 4: INK 0: PRINT AT 13
"Desee intentar de nuevo?"
730 SOUND 7,30
740 SOUND 0,68,1,3,8,12
750 PAUSE 0
760 SOUND 2,151,3,2,9,12
770 PAUSE 0
780 SOUND 4,45,5,2,10,12
790 PAUSE 100
800 SOUND 0,0,1,0,2,0,3,0,4,0,5
810 LET a$=INKEY$. IF a$="s" OR
a$="S" THEN GO TO 110

```

```

820 IF a$="7" AND a$<"N" THEN
GO TO 100
830 STOP
840 BEEP .25,30 IF b(1)=1 THEN
GO TO 770
850 LET b(1)=1 GO SUB 860
860 b(1)=b(1)+1
870 GO SUB 880: IF b(1)=1 THEN
GO SUB 880
880 PRINT AT 21,22 PAPER 4 IN
K 0 " " GO TO 800
890 LET c=0 FOR I=1 TO 7 LET
a(1,I)=1 NEXT I: LET c=1
900 PAPER 4: INK 0: PRINT AT 11
,3 PAPER 4: " "
910 SOUND 7,62,8,15
920 FOR I=50 TO 100
930 SOUND 0,1: PAUSE 3
940 NEXT I
950 SOUND 6,6,7,7,8,16,9,16,10
,16,13,8
960 PAUSE 0
970 SOUND 0,0,9,0,10,0
980 GO TO 800
990 LET w=INT ((a-1)/13)
1000 LET a$="♥" AND NOT w+("♦"
w=1+("♠" AND w=2)+("♣" AND
w=3)
1010 INK 2 IF w>1 THEN INK 0
1020 LET w=a-w*13
1030 IF w=10 AND w<>1 THEN LET b
=INT a$ w RETURN
1040 LET b$=(("♠" AND w=10)+("♥"
AND w=11)+("♦" AND w=12)+("♣" AN
d w=13)+("A" AND w=1) RETURN
1050 LET a=b: GO SUB 800: PRINT
AT 9,29 PAPER 4: " " PRINT AT
10,29 " "AT 7,29,a$: " "
1060 PRINT AT 8,29 " "AT 9,29
" "a$": AT 10,29 " "b$": RETUR
N
1070 PAPER 4: FOR I=12 TO 16: PR
INT AT 1,29 " " NEXT I: PAPER
RETURN
1080 PAPER 2: INK 7 FOR I=12 TO
16 PRINT AT 1,29 "FFF" NEXT I
PAPER 7: RETURN
1090 FOR d=0 TO 7
1100 INPUT file: POKE USA "E"+d,
file
1110 NEXT d

```



TREN

DE MULTIPLICACION

Es un programa educativo que trata a través de un juego a comprobar que Ud. sabe multiplicar. Lo principal es tomar el tren al final de la vía dando respuestas correctas al problema de multiplicación que se genera al azar. Para comenzar el juego presione RUN y ENTER, la máquina preguntará por el nivel de dificultad que Ud. desea. Cuando el tren alcance el final de la vía, aparecerá el cartel "Bien hecho". Las alternativas de dificultad pueden ser modificadas, cambiando de la línea 50 a la 80.

NOTA GRAFICA:

Línea 100- Shift R- 3 espacios-
Shift B- 1 espacio.

Línea 110- Shift B- 3 espacios-
Shift 5.

Línea 120- Shift R- Shift
Espacio- Shift R- Shift E.

Línea 130- 14 quiones.

```

10 LET Y=10
20 LET X=0
30 PRINT AT 3,0;"MANIA 1  OR 2"

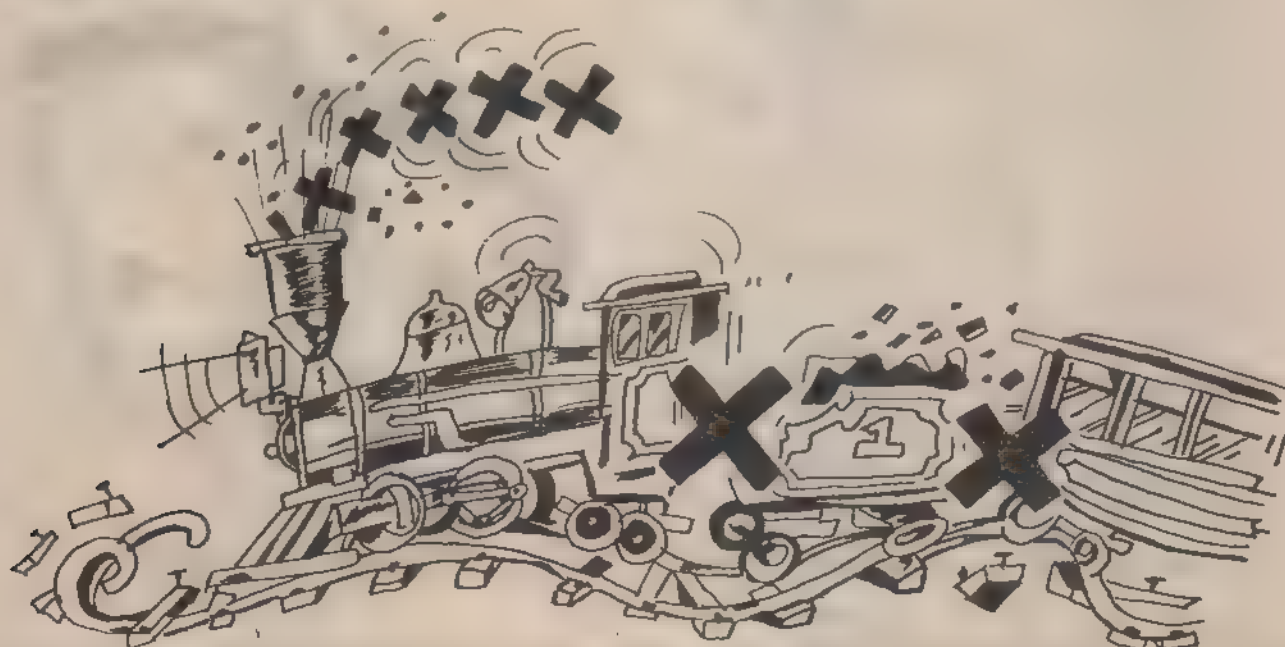
40 INPUT "A$";
50 LET A=INT (RND*20)
60 LET B=INT (RND*20)
70 LET C=INT (RND*1)
80 LET D=INT (RND*10)
90 CLS
100 PRINT AT Y,X;" "
110 PRINT AT Y+1,X;" "
120 PRINT AT Y+2,X;" "
130 PRINT AT 13,0;"-----"
140 PRINT AT 0,0;"TOMAR AL FINA"
150 IF A$="1" THEN PRINT AT 5,0
160 IF A$="2" THEN PRINT AT 5,0
170 LET H=A*B
180 LET K=C*D
190 INPUT "Z";
200 IF A$="1" AND Z=H THEN GOTO 240
210 IF A$="2" AND Z=K THEN GOTO 240
220 PRINT AT 15,0;"ERRONEO"
230 PAUSE 100

```

```

240 GOTO 50
250 PRINT AT 9,X+2;" "
260 LET X=X+1
270 IF X>10 THEN GOTO 300
275 PRINT AT 12,15;" "
280 PRINT AT 13,15;"BIEN HECH"
290 STOP
300 PAUSE 150
310 GOTO 50

```



ENUNCIADO GO TO

El siguiente es un programa que calcula el producto y el cociente de dos numeros, e imprime el resultado.

```
10 READ B,C
20 LET A1=B*C
30 PRINT "EL PRODUCTO ES"
    A1
40 LET A2=B/C
50 PRINT "EL COCIENTE ES"
    A2
60 DATA 36,9
70 END
```

Tal como esta el programa, cada vez que queramos repetir el calculo para un nuevo par de numeros, tendremos que volver a correr el programa con un nuevo enunciado DATA. Seria conveniente que pudieramos hacer que la computadora repitiera el calculo con otras cantidades, sin nuestra intervencion. Si insertamos la linea

```
55 GO TO 10
```

podemos hacer que el flujo vuelva a la linea 10 cuando llegue a la 55, y vuelva a empezar la secuencia del programa con

```
10 READ B,C
```

Si luego modificamos el enunciado DATA:

```
36 DATA 36,9,20,5
```

en la segunda corrida la maquina leera para B y C, respectivamente, los valores 20 y 5 e imprimira el segundo grupo de resultados. La maquina repetira los lazos de retorno por todo el programa hasta agotar la lista de numeros que le dimos en el enunciado DATA, y el programa terminara.

Asi pues, "GO TO" equivale a una bifurcacion o transferencia en la secuencia de instrucciones (cambia la secuencia normal de ejecucion).

ENUNCIADO FOR

El ciclo FOR es una orden para repetir un conjunto de enunciados para cada valor del contador, en el rango especificado. Estos ciclos se controlan contandolos, pues se sabe de antemano la cantidad de repeticiones necesarias.

```
10 FOR I=1 TO 10
20 PRINT I
30 NEXT I
40 END
```

"I" se denomina contador de ciclo y puede ser cualquier variable sin subindice.

Este enunciado FOR ordena la ejecucion de todos los enunciados siguientes hasta el enunciado NEXT, para I=1, I=2, etc., hasta I=10. El enunciado NEXT I abarca todos los que seran ejecutados y repetidos para cada valor de I. Siempre tiene que haber un enunciado NEXT correspondiendo a un enunciado FOR.

Despues del signo igual en un enunciado FOR hay que especificar un valor inicial del contador de ciclo. Despues de la palabra TO, hay que dar el valor final del contador de ciclo. Tanto el valor inicial como el final, pueden especificarse como una constante con signo, o sin el, o como una expresion.

Se supone que el incremento del contador de ciclo es de uno. Si se desea un incremento distinto de uno, debera especificarse a traves de un STEP. Estan permitidos STEP negativos (cuando queremos decrementar el contador).

```
10 FOR I=1 TO 10 STEP 2
20 PRINT I
30 NEXT I
40 END
```

```
10 FOR I=1 TO N/2 STEP -1
20 PRINT I
30 NEXT I
40 END
```

Si el valor inicial es mayor que el final y el incremento es positivo, o si es menor que el final pero el incremento es negativo, no se ejecutara el ciclo.

El siguiente enunciado que se ejecutara, en tales casos, sera el que siga al enunciado NEXT correspondiente.

Quizá Ud. este pensando en incluir un ciclo FOR-NEXT dentro otro; es decir, anidar ciclos (ponerlos unos dentro de otros). Esto es posible solo si se lo hace en forma correcta: no deben enlazarse ni superponerse en forma parcial y nunca debe quedar solo un enunciado FOR, dentro de un ciclo exterior mas grande, sin su NEXT correspondiente. El enunciado FOR siempre debe estar

acompañado por su correspondiente enunciado NEXT, dentro del ciclo mayor.

```
10 FOR X=1 TO 4
20 PRINT X
30 FOR Y=1 TO 3
40 PRINT Y
50 NEXT Y
60 NEXT X
70 END
```

ENUNCIADO IF

Las decisiones en BASIC, son tomadas a traves del enunciado IF.

Este consta de tres partes: la palabra IF (el si condicional) que identifica el enunciado, una relacion logica que puede ser verdadera o falsa, y la palabra THEN (entonces) seguida de la accion a tomar cuando la relacion tenga el valor logico "verdadero".

Las relaciones que pueden usarse son:

```
= igual a
<> no es igual a
< menor que
<= menor o igual a
> mayor que
>= mayor o igual a
```

Algunos ejemplos:

```
40 IF B-4*A>A-C/2 THEN GO ... TO 200
30 IF D=3 THEN LET C=A+B
10 IF T<>4 THEN LET A=B*T
80 IF A*C<D THEN GO TO 50
```

Las dos aplicaciones mas usuales del enunciado condicional IF son: 1) crear una bifurcacion hacia otra parte del programa como resultado de una prueba (instrucciones 40 y 80).

2) incluir o excluir un enunciado intermedio, basandose en el resultado de una prueba (instrucciones 30 y 10).

EJERCICIOS

- 1) Hallar el mayor de un conjunto de 20 numeros e imprimirlo.
- 2) Realice la entrada y comprobacion de eco de dos valores, b y c, y halle la raiz de la ecuacion $bx+c=0$.
 - i) Si $b=0$ y $c<>0$, imprima la salida " $bx+c=0$ no tiene raiz".
 - ii) Si $b=0$ y $c=0$, imprima la salida "todo numero real satisface la ecuacion".
 - iii) Si $b<>0$, imprima la salida "la raiz de $bx+c=0$ es", seguida de la raiz.

SOLUCION DE LOS EJERCICIOS DE LA EDICION ANTERIOR

```

1) 10 READ R
    20 DATA 2
    30 PRINT AT1,1;"RADIO",
        AT1,11;"PERIMETRO"
        AT1,22;"AREA"
    40 LET A=2*PI*(R^2)
    50 LET CIR=2*PI*R
    60 PRINT AT3,1;R,
        AT3,11;CIR,
        AT3,22;A
    70 END

```

```

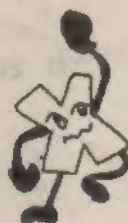
2) 10 INPUT A,B
    20 LET X=A*B
    30 LET Y=A/B
    40 PRINT AT8,10;"X=";X
    50 PRINT AT8,20;"Y=";Y
    60 END

```

TK-83/ TK-85

TS 1000/1500

CLASIFICACION ALFABETICA



```

9500 FAST
9530 LET A=1
9540 LET N=100
9550 IF 2**A>N THEN GOTO 9580
9560 LET A=A+SGN A
9570 GOTO 9550
9580 LET F=2**A-1
9590 LET F=INT (F/2)
9600 IF NOT F THEN GOTO 9750
9610 LET D=N-F
9620 LET B=SGN A
9630 LET A=B
9635 LET E=A+F
9650 IF A$(A)>A$(E) THEN GOTO 97
9660 LET B=B+SGN B
9670 GOTO (B>D AND 9590)+(9620 A
ND B<=D)
9700 LET B$=A$(A)
9710 LET A$(A)=A$(E)
9720 LET A$(E)=B$
9730 LET A=A-F
9740 GOTO (9650 AND A(1)+(9630 A
ND A>=1))
9750 SLOW
9760 CLS
9770 FOR F=1 TO 100
9780 SCROLL
9790 PRINT A$(F)
9800 NEXT F
9810 RETURN

```

Es una rutina que nos permite ordenar A\$ en forma alfabética. Altere las líneas 9540 y 9770 de acuerdo con el número de elementos que tenga A\$. Por ejemplo si Ud. dimensiona A\$ como A\$(100,15) el número en ambas líneas es 100.



PUBLICACION

SINTAX

CASILLA DE CORREO Nº 641

1900 - LA PLATA - BUENOS AIRES